89 Fifth Avenue, 7th Floor

New York, NY 10003

www.TheEdison.com

212.367.7400

HPC Workload Management Tools: A Competitive Benchmark Study

Comparing IBM Platform LSF, Open Grid Scheduler, SLURM and TORQUE for Throughput and Agility

Printed in the United States of America

The information contained in this document is based on IBM provided materials and independent research and was aggregated and validated for Edison Group, Inc. by the Edison Group Analyst team.

First Publication: July, 2014

Produced by: Matthew Elkourie, Analyst; Manny Frishberg, Editor; Barry Cohen, Editor-in-Chief

Table of Contents

Executive Summary

High Performance Computing (HPC), a mature industry operating within numerous verticals, is currently experiencing renewed focus and growth as mainstream applications increasingly have similar requirements for compute resources and management of increasingly complex HPC infrastructure. It makes sense then, that along with HPC industry growth, a renewed interest in cluster management and job scheduling of Message Passing Interface (MPI) and batch workloads would also be receiving renewed interest and focus.

High Performance Throughput Computing (HPTC) takes the core principles of traditional HPC and emphasizes throughput and agility. When an HPTC infrastructure is performing at its best, the results obtained by a highly agile, low latency system lead to more work in less time, with less infrastructure and a faster turnaround to market.

It is clear that IBM® Platform™ LSF® drives 150x greater throughput while being 2800x more responsive to changing workloads, translating into reduced delays, more work being performed, and better responsiveness to changing customer priorities.

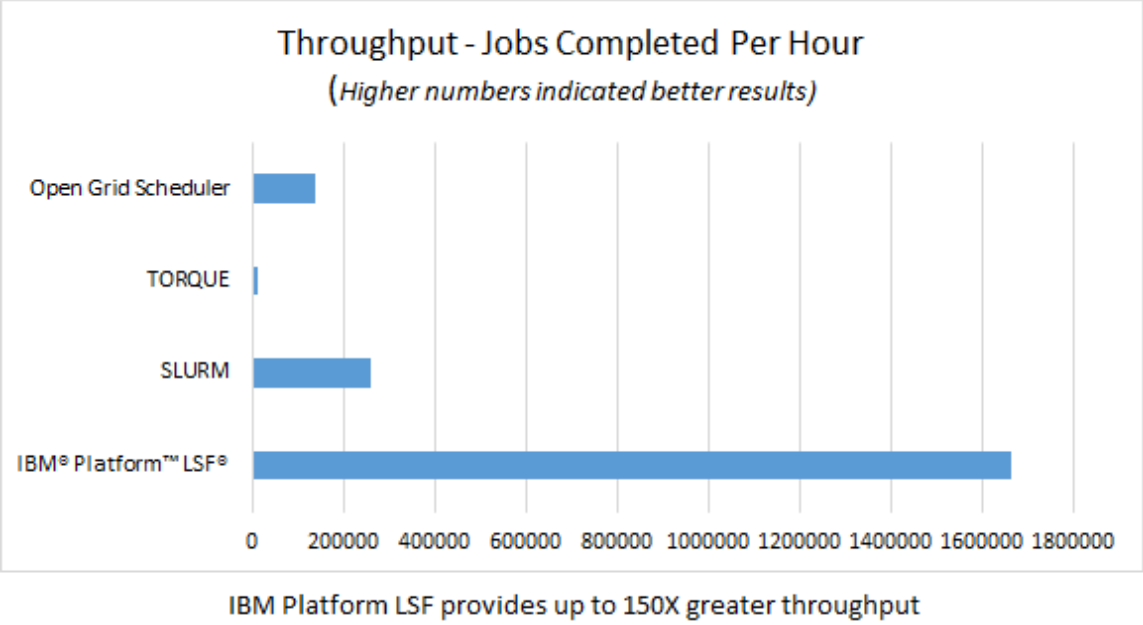

Figure One: IBM Platform LSF Demonstrates Throughput Dominance[1]

[1] For benchmark details see the "Benchmark Validation and Competitive Analysis" section starting on page 11 of the white paper.

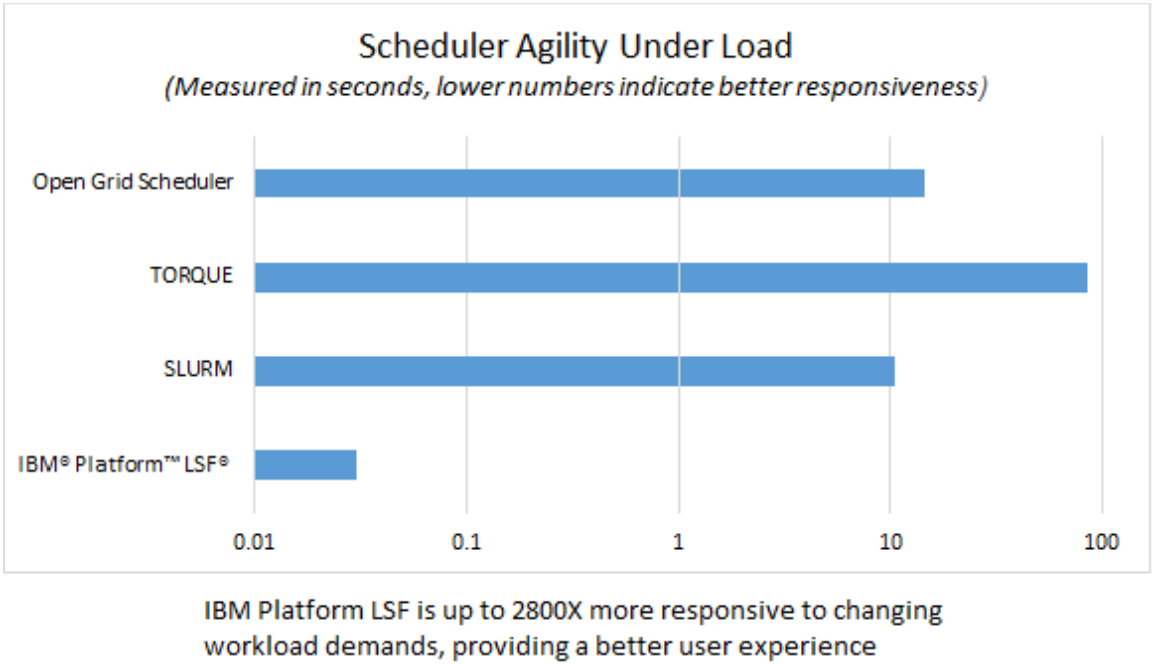

Figure Two: IBM Platform LSF Demonstrates Scheduler Agility Dominance[2]

IBM Platform LSF brings together the robust feature sets, management capability, speed, and reliability that is demanded by both administrators and consumers while delivering agility and throughput that is industry leading. While the field of HPTC has grown in size and scope, IBM Platform LSF has kept pace in its ability to scale and meet the demands of systems administrators and end users.

[2] For benchmark details see the "Benchmark Validation and Competitive Analysis" section starting on page 11 of the white paper.

Key Findings

During the course of reviewing and evaluating the results of benchmark data generated by IBM Platform LSF along with competitive technologies, it quickly became clear that IBM Platform LSF consistently performed faster, and produced quicker results than the similar competitive technologies.

Workload Manager	Throughput[3]	Agility[4]
IBM Platform LSF	1,666,666	0.03 Seconds
SLURM	258,435	10.43 Seconds
TORQUE	11,000	84.93 Seconds
Open Grid Scheduler	138,089	14.63 Seconds

Figure Three: Key Findings Summary – Throughput and Agility

Additionally, the IBM Platform LSF benchmark results excelled in tests of system responsiveness time, as well as when performing concurrent queries while under heavy workloads.

Key findings include:

- Demonstrating throughput capability, in Benchmark 1, IBM Platform LSF performed jobs up to 150x faster (a ratio of 150:1) over the other tested products, measured in jobs completed per hour. In environments where large amounts of short-running jobs are being executed, achieving the ratio of performance benchmarked, customers can take advantage of faster time to market strategies, or possibly better quality products as a result of additional verification and simulation time as a result of fast job completion.
- In Benchmark 1, IBM Platform LSF completed more jobs than the other tested products combined, in the period of 1 hour. Based on the benchmark results, customers could realize accelerated results retrieval from jobs submitted and reduction in time needed to obtain job results per job submission cycle due to increased throughput.

[3] Measurement of jobs completed in 1 hour where a higher number of jobs completed is the desired result.

[4] Measurement of scheduler responsiveness under load, demonstrating platform agility in response to potential workload changes or user input while the system is under load. A lower number is a desired result.

- In Benchmark 2, IBM Platform LSF job query responsiveness under load was up to 250x faster than the other tested products, and responsiveness was always under 1 second. The agility displayed by IBM Platform LSF to consistently perform equates to more work being completed in less time, with reliable results providing advantages in changing job prioritization, reducing delays and better responding to changing customer priorities that leads to an overall better end-user experience.
- In Benchmark 2, we see that IBM Platform LSF was up to 2,800x faster in dispatching a high priority workload once submitted consistently, leading to more work being performed while requiring far less time spent in submitting jobs.
- In Benchmark 2, IBM Platform LSF excels in throughput dominance, executing concurrent queries at the rate of nearly 4x the number of queries performed in the same time period as the other, similar tested products. Once again, being able to complete queries more efficiently allows operators and end users to focus more on the results, with less time spent on retrieval and management of their results.

Introduction

Objective

This report examines and validates two typical HPTC styled benchmarks, performed by IBM, and the resulting information related to run time and number of jobs, as compared to job execution time. In addition, consideration was given to required installation of utilities and tools in order to perform the benchmarks, and the time and effort required to achieve baseline results.

Since HPTC jobs are highly varied and dependent on objectives unique to the end-user community, attention was focused on not only the results of standardized testing, but also on the ease in which each environment was created. The hardware environment of the System Under Test (SUT) was maintained by keeping the systems and related infrastructure the same from runs of the same test samples, while using the different engines evaluated.

Of the competitors evaluated, the goal was to digest and evaluate the consistency of tools across the varying job submissions, to identify the engine(s) with the best consistency of results, and to evaluate the effort required to achieve those results.

Audience

Decision makers and evaluators involved in Electronic Design Automation (EDA), Life Sciences, Computational Fluid Dynamics (CFD), Seismic Tomography, and similar verticals will be interested in the results obtained. Additionally, target audiences include inter-departmental systems managers challenged with resource availability, cluster maintenance, and performance constraints where resources would best benefit from a unified HPTC management suite.

HPTC and IBM Platform LSF

High performance technical computing, along with the associated disciplines and requirements often associated with business or scientific inquiries, is expanding into new areas of study and exploration.

In the fields of EDA, Life Sciences, CFD Analysis, Seismic Tomography, and other related fields of study and research, HPTC plays an ever increasing role of providing answers to researchers. By leveraging common, well-established, and proven applications and infrastructure, scientists and researchers alike continue to increasingly find new and exciting ways to leverage high performance HPTC infrastructures and platform managers.

As HPTC has evolved, IBM Platform LSF continues to provide the consistency, high performance, and accurate results demanded in industries where time literally equals revenue won or lost. So too, the ability to cohesively present unified and consistent interfaces to engineers, researchers, and scientists has evolved, while keeping pace with ever increasingly complex and dense system architectures.

IBM Platform LSF is extensible, modular, and well supported on one of the largest playing fields of system architecture available. Whether deploying on a traditional x86 platform, or legacy or less common architectures, IBM Platform LSF delivers results. Although the testing in this white paper will focus on x86 architecture, Platform LSF provides support on alternate platforms such as IBM® POWER®, Intel® Itanium®, Oracle® SPARC®, and ARM®.

In this white paper, the benefits IBM Platform LSF brings to the table for HPTC jobs will be examined. From performance metrics showing platform resiliency and throughput, to the ability of LSF to outperform its peers in raw latency and execution capabilities, this product should be at the top of any review where investment into infrastructure and management of often highly complex environments is key.

HPTC Performance Scalability: The Challenge

HPTC as a platform requires more than just the act of procurement and investment in hardware and software tools and licenses. A successful HPTC environment requires platform administrators with the abilities to correctly maintain and monitor the platform, install and configure related software and third-party tools, and to ensure that there is uniformity in not only software and license availabilities, but also in accomplishing the above tasks in such a way that end-users are able to schedule and continue to perform their end-user tasks with minimal to no workload interruption or scheduling inefficiencies.

HPTC clusters are expected to perform as quickly and efficiently as possible, while at the same time offering users an environment that is agile yet responsive throughout the user interaction process. Whether users are actively submitting jobs or batches, reviewing job results or monitoring cluster health for their resources, or interactively designing future tasks to be submitted, one of the unique challenges in cluster design and performance is to ensure that users can complete their tasks quickly and efficiently, while not affecting job runs or queued job performance.

As an administrator, the job of maintaining an HPTC cluster can be extremely difficult. Decisions that are made in a fluid, working environment can span duties such as:

- Troubleshooting cluster environment performance problems
- Adding and maintaining dozens of different software packages
- Optimizing cluster specific or batch specific code for jobs
- Expanding an existing cluster
- Servicing end-user requests for hardware or software specific to a non-standard job submission requirement

Competitors to Platform LSF: Overview

The benchmark test results and scheduler features examined in this white paper were evaluated against what were viewed as popular alternatives, in terms of features and capability, to IBM Platform LSF. Sample test runs using similar parameters were evaluated against Simple Linux Utility for Resource Management (SLURM), Open Grid Scheduler, and Terascale Open-source Resource and QUEueManager (TORQUE). The SUT units were installed with, and tests performed under Linux running on standard IBM x86 nodes.

SLURM, an open-source workload manager, offers free availability (download) of the SLURM stack, in addition to commercial level support from SchedMD, the company that develops, distributes, and maintains SLURM. SLURM supported platforms are currently limited to Linux®, Berkeley Software Distribution (BSD®) derivatives, Apple® Mac® OS X®, IBM® AIX™, and Oracle® Solaris®. SLURM supports x86 infrastructure, and provides support for several unique architectures.

TORQUE is an open-source resource and workload manager, available via download from Adaptive Computing. While free, TORQUE commercial support is available. TORQUE is based on the original Portable Batch System (PBS) project, and with the inclusion over time of over 1200 patches, has incorporated many advancements, such as scalability and fault tolerance enhancements. TORQUE is typically used in conjunction with either Maui or Moab for improved utilization, scheduling, and administrative tasks.

Open Grid Scheduler (OGS), based on the SGE or Sun Grid Engine®, is a job scheduling, dispatch and management engine. While the focus of the testing is based on the OGS version of the engine, there have been other uses for the solution. A commercial support option for the engine currently is maintained by Univa®, who also distributes the engine as Univa Grid Engine®.

While some of the offerings reviewed had unique or niche focused tool or platform-specific advantages, overall the IBM Platform LSF offerings and toolsets provided the most seamless, feature-rich, integrated approach for typical and scalable HPC operations. While Platform LSF is scalable to the largest cluster sizes, LSF management features and out-of-box tools provide a cohesive platform for IT departments regardless of size, and without extensive integration of third-party toolsets.

IBM Platform LSF: Deep Dive

IBM Platform LSF (also known as LSF) is a workload management platform and job scheduler engine for use typically in distributed HPTC environments.

LSF operates under a variety of different available underlying hardware architectures. For example, IBM Platform LSF runs on the following architectures:

- IBM POWER and IBM System z® series
- Intel and AMD® x86-64
- Intel Itanium
- Oracle SPARC
- ARM

Cluster environments require a delicate balancing of available hardware resources, available time shares on the cluster for jobs to be submitted and executed. There are great challenges to managing a variety of different third-party software, as well as the challenge of certain jobs requiring what can be the same software but a different version installation requirement, within the same cluster that other end-users are accessing.

In addition to the onboarding process of designing and deploying/servicing an HPTC cluster, there is a real challenge to then optimizing the platform via job scheduling software and batch submission control software. Such a platform must be expected to perform tasks in a repeatable fashion, ideally with expected, non-varying consistency with regards to performance.

IBM Platform LSF is designed to tackle the tough IT challenges that HPTC computing commands, by dramatically reducing the time investment operators need to spend crafting jobs, while offering increased execution time and low latency access to running jobs and performing queries.

In addition to typical feature sets found in other schedulers, Platform LSF delivers enterprise level policies and features including, but not limited to:

- SLA (Service Level Agreement) based scheduling
- Scheduling of high throughput jobs and priority balancing of varying jobs within a multi-tiered infrastructure

- Multi-cluster scheduling
- Energy-aware scheduling

IBM Platform LSF is feature-rich, providing additional features and services by way of software add-ons that enhance the core features of Platform LSF. A product matrix is provided in Appendix A – Additional Software Add-ons.

Benchmark Validation and Competitive Analysis

Benchmark 1

Environment

The job submission consisted of a configuration of 16 IBM® System x™ iDataPlex™ dx360 M3 servers and a single head node, configured as follows.

Hardware:

- 2x Intel® Xeon® X5670 CPUs, running at 2.93Ghz
- 24GB of RAM
- 1x 146GB (SAS 15k RPM) IBM ESXS series drives
- 1GBe network adapters

Network:

- Standard 1GBe (Gigabit) Adapters, single connection between head nodes and compute nodes, no networking bonding or additional adapters employed
- Standard network configuration, the use of hardware compression and Jumbo Frames was not used

Storage:

- Local disk storage was configured with IBM® Elastic Storage, version 3.5. The use of a SAN was not utilized. The workload managers were installed to the GPFS file system

Operation System (OS):

- Red Hat Linux, version RHEL 6.3 was installed. Standard OS installation method used, with no additional packages installed other than MUNGE (An authentication service for creating and validating credentials)

The engines under review are IBM Platform LSF, SLURM, TORQUE, and Open Grid Scheduler. Additional detail regarding the engines used can be found in the "Competitors to Platform LSF: Overview" section of the paper, and additional versioning information can be found in Appendix C – Workload Manager Versions Tested as well as the IBM provided Workload Manager Configuration Files[5].

[5] https://ibm.biz/BdFfhM

Methodology

Benchmark 1 was designed to test raw throughput of short running jobs through a workload management system. The target environment is a 17 node cluster made up of a dedicated workload management head node, and 16 compute nodes, each equipped with 8 job slots (meaning that 8 serial jobs can run at one time on a given node). This made for a total of 128 job slots.

Using the appropriate commands from each respective workload manager, high-level benchmark scripts performed the following:

1. Configure 8 job slots per node (16 nodes * 8 slots total)
2. Disable scheduling of workload (disable queue, etc.)
3. Submit 100000 /bin/sleep 0 jobs to the respective workload manager
    a. Standard error, output directed to /dev/null for jobs
    b. Job submissions initiated from /tmp
4. Enable scheduling of workload
5. Benchmark script will print out a throughput value (# jobs / hour)

Analysis

Edison Group validation of the benchmark results was performed independently of IBM, and consisted of review in the testing methodology employed, the configuration of the test harness configuration files, and finally review of the results as produced by the SUT.

Source files for the benchmark runs and workload managers were scrutinized for configuration detail and validation as to the configuration, ensuring that results obtained were consistent with the environment presented to each workload manager.

Test Results

Benchmarks were performed over a series of three runs per workload manager and are illustrated below in the summary graph. Tables representing the raw data used to calculate the graph results are included in Appendix B – Raw Test Results.

A review of the results and source documents as well as source files, along with discussion of findings with the researchers responsible revealed that Platform LSF was

able to perform an order of magnitude faster, in terms of jobs completed per hour, using default configuration settings.

Research and prior experience by Edison Group's analyst agree with statements by the project researchers that further effort in configuration, while in some cases significant to perform, may have enhanced some of the achieved test results. As the subject of this analysis is performance out of the box, without requiring end-users to devote extensive time to perform pre-configuration, no additional effort was expended in further performance enhancement.

It should be further noted that the results obtained did not leverage readily available, high speed and/or lower latency network interfaces or mechanisms. Performance results indicate that additional network capacity would have served to allow more jobs to be completed, as the constraint observed in this test clearly indicate that lack of additional network capacity is indeed a limiting factor. While the hardware platform evaluated is not configured with the absolute latest in available processor technology or maximum memory configuration, in Benchmark 1, system resources themselves were not an evident limiting factor.

An illustrative output of the results obtained in Benchmark 1 is provided below. In this table, higher numbers equate to a higher performance profile and illustrate the amount of jobs able to be completed per hour, using the metrics exposed in the Environment section of this paper under Benchmark 1.

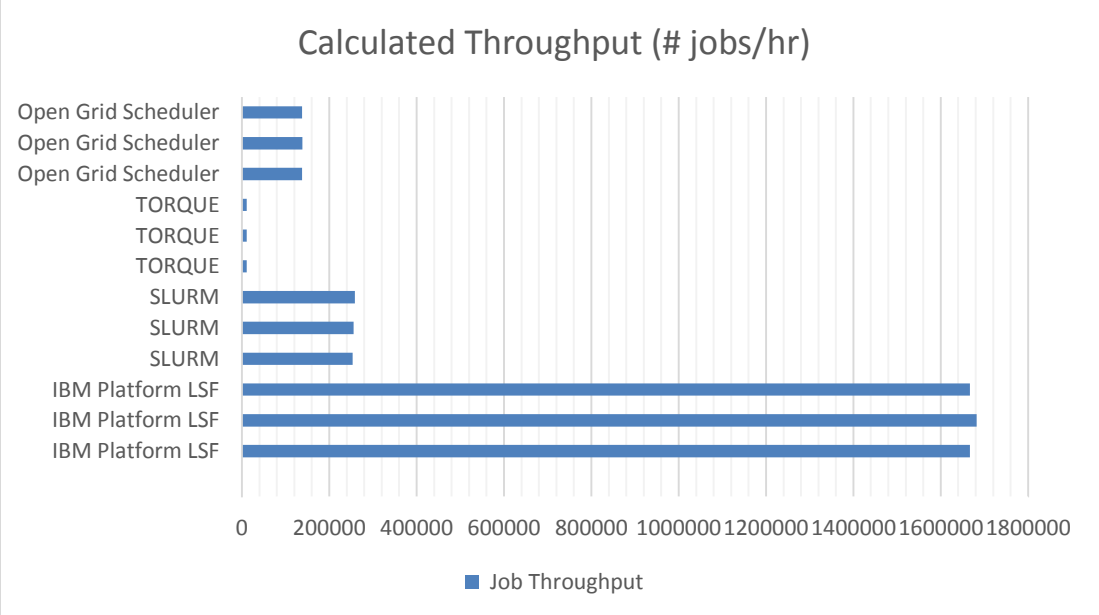

Figure Four: Benchmark 1 Testing Data - Throughput

Benchmark 2

Environment

The job submission consisted of a configuration of 64 IBM® System x™ iDataPlex™dx360 M3 servers and a single head node, configured as follows.

Hardware:

- 2x Intel® Xeon® X5670 CPUs, running at 2.93Ghz
- 24GB of RAM
- 1x 146GB (SAS 15k RPM) IBM ESXS series drives
- 1GBe network adapters

Network:

- Standard 1GBe (Gigabit) Adapters, single connection between head nodes and compute nodes, no networking bonding or additional adapters employed
- Standard network configuration, the use of hardware compression and Jumbo Frames was not used

Storage:

- Local disk storage was configured with IBM® Elastic Storage. The use of a SAN was not utilized. The workload managers were installed to the GPFS file system

Operation System (OS):

- Red Hat Linux, version RHEL 6.3 was installed. Standard OS installation method used, with no additional packages installed other than MUNGE

Benchmark Specifics:

- Cluster comprised of 65 nodes in total:
  - 1 x scheduler master node (no workload permitted to run)
  - 62 x compute nodes
  - 2 x client nodes (query load generation)
- Allocate 1 compute node to the high priority admin queue and the remaining 61 compute nodes to the normal queue
- 2 dedicated client hosts each run 10 concurrent job query commands

As with benchmark test one, the engines being reviewed are IBM Platform LSF, SLURM, TORQUE, and Open Grid Scheduler.

The benchmark submitted 100,000 single jobs with varying resource requirements to the normal queue. Leveraging the same test user, the cluster was accessed from 2 different service hosts, with 10 job queries executed continuously. The objectives were to determine the job query time, job submission time, and high priority job response time in terms of seconds. A lower value number is considered to be indicative of better response time.

Methodology

Benchmark 2 is focused on the performance of the job scheduling and system responsiveness under a large job load. The job submission time, job query time and system response time to a high priority job will be measured.

At a high-level the benchmark scripts performed the following (using the appropriate commands from each respective workload manager):

1. Submit 100K single jobs with 1000 different resource requirements (e.g. mem>i, i=1, 1000) to the normal queue with the normal queue inactivated, the job runtime should be a mix of 5 – 120 seconds, measure the total time spent
2. Launch 10 concurrent job queries continuously running from the same test user on 2 different server hosts. The return code of each job query is recorded
3. Activate normal queue
4. Launch a script with below operations on master host and keep it running for 30 minutes:
    a. Submit a short simple job (e.g. sleep 1) to the high priority queue (e.g. admin) and record the time spent by the command
    b. Wait for 60 seconds, and then query the admin queue and record the time spent by the command
    c. Go back to step a
5. Get the average job submission time based on the data recorded
6. Get the average job query response time based on the data recorded
7. Get the average high priority job response time based on the accounting files/information for the jobs submitted to the high priority queue “admin”

8. Get the total number of successful and failed job query commands

Analysis

As with Benchmark 1, Edison received data from three test runs (raw and summarized), in addition to the physical and software makeup of the SUTs. Further, the configuration files were analyzed for consistency in the job makeup submitted to the SUTs.

By comparing the source raw data of each engine and the resultant output of each test run generated by each engine, data was collected and is displayed in Tables 2-5, in Appendix B – Raw Test Results similar to Benchmark 1 (Table 1), respectively.

Test Results

Benchmark 2 was aimed at several criteria.

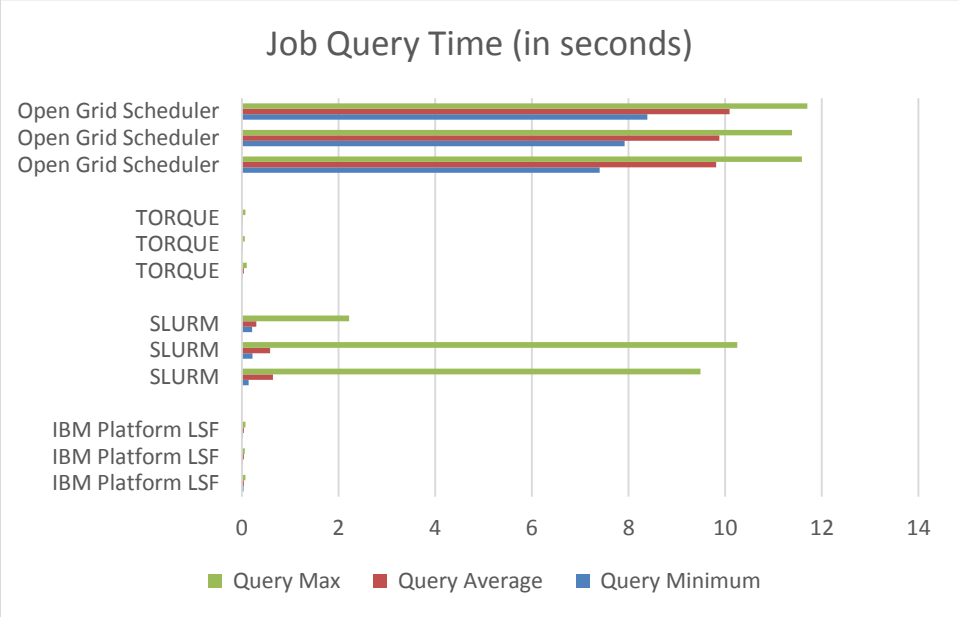

Figure Five: Benchmark 2 Testing Results – Job Query Time

As illustrated in the above graph, first, responsiveness while under a heavy workload was measured, in terms of responsiveness at the console.

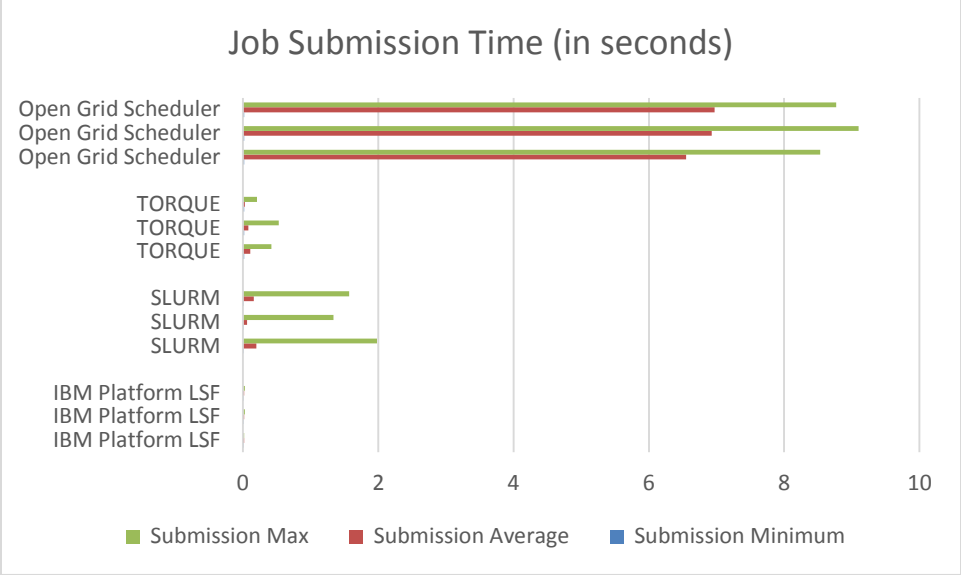

Figure Six: Benchmark 2 Testing Results – Job Submission Time

Second, submitting high priority jobs while other jobs are running and capturing submission time was gauged. The accompanying graph above illustrates over three separate runs the time needed to successfully submit the jobs.

This is critical when time to complete jobs is of utmost importance. Scaled over a large user base, backups in jobs waiting to be submitted to the queue can develop where engine sluggishness is observed.

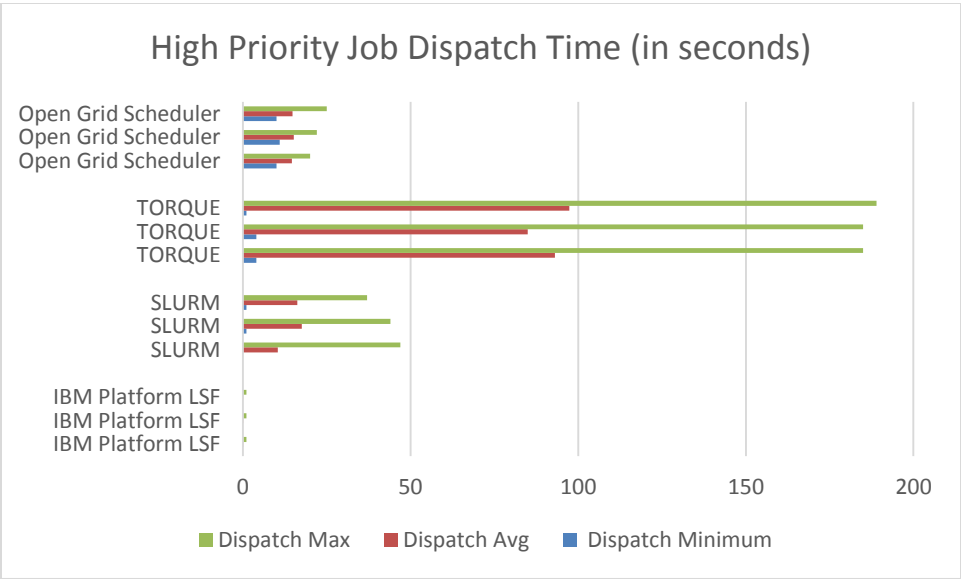

Figure Seven: Benchmark 2 Testing Results – High Priority Job Dispatch Time

Third, the time to dispatch high priority jobs while other jobs are running was gauged. The accompanying graph above illustrates over three separate runs the time needed to dispatch the jobs.

This is critical when time to complete jobs is of utmost importance. Scaled over a large user base, a backup in jobs waiting to be dispatched to the queue can develop, causing a significant decrease in work productivity.

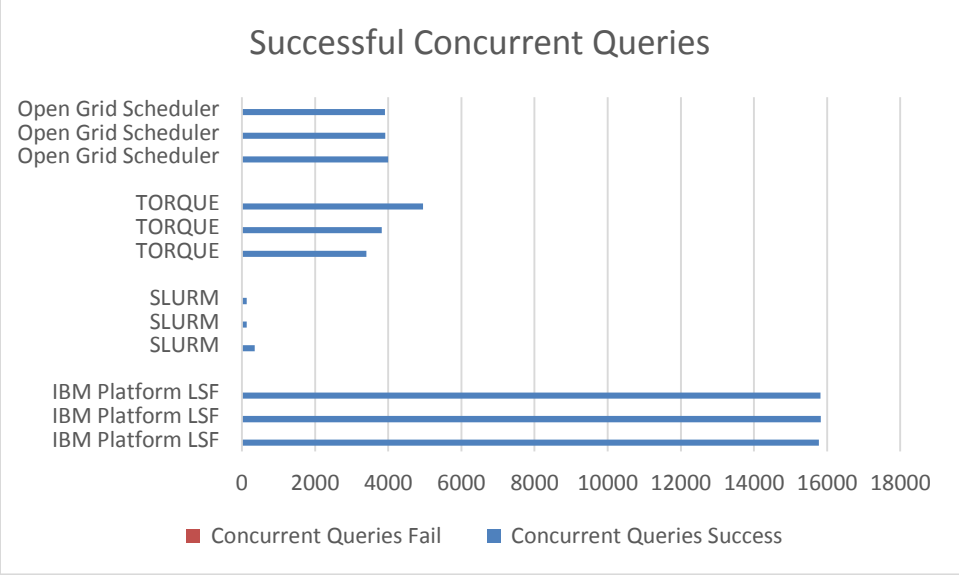

Figure Eight: Benchmark 2 Testing Results – Successful Concurrent Queries

Finally, the number of successful, non-failing queries was evaluated and gauged. As illustrated above, larger amounts of successful queries performed in the test is preferred, and is meant to simulate users checking in on jobs running on a busy cluster.

As was noted in the previous benchmark, the environment into which the jobs were dispatched is not proprietary or necessarily staged to elicit results or provide favoritism amongst the engines and SUTs. Edison Group evaluated each job or batch script submitted, and verified that the environment is equalized to provide a fair, consistent environment from which valid data points and results could be gathered. Further, while some engines have the ability to scale even further or faster with additional third-party or commercial add-ons, in the case of this testing, baseline performance was gathered to set a typical bar without use of performance enhancing add-ons.

Benchmark 2 scoring is measured in terms of seconds, where a lower score is a more desirable score. Keeping in mind that user interactivity is the focus of the benchmark, with some emphasis being placed on overall system responsiveness, scoring therefore should consider that excessive timing in addition to adding delay to projects or job submissions also can have influence with regards to user's assumptions of overall cluster performance, real or perceived.

Conclusions and Recommendations

HPTC benchmarking is a large field in which a researcher can spend an immense investment in time and due diligence. Over the last few months, Edison Group's team investigated the general state of HPTC, the several verticals in IT that leverage HPTC, and metrics that illustrate meaningful impact to efficiency and productivity.

Each of the platforms evaluated (Platform LSF, SLURM, TORQUE, Open Grid Scheduler), while sharing some commonality in function, exhibited unique characteristics that thorough benchmarking was able to clearly illustrate. In our research, we focused on user experience and system responsiveness as a measure of efficiency and ease of work performance. Ensuring that jobs can be completed swiftly, accurately, and in a repetitive and predictable manner can be key to whether a department or research lab is able to retrieve data for time-sensitive deadlines.

As illustrated in Benchmark 1, the focus of our research point was to establish throughput as measured by short running jobs, where literally thousands of jobs were being executed. The results are meaningful, as being able to produce quick results with accuracy allows researchers to adhere to tight schedules and condense workloads. The additional work being performed without errors in a given time frame allows for time on the cluster to be reallocated to other researchers and users of the system, thus increasing the value of the platform on which work is being executed, and likely increasing Return on Investment (ROI) through increased productivity.

In Benchmark 2, we shifted focus to system agility or responsiveness while under duress. Often times, HPTC systems have multiple loads being scheduled, where agility determines the ability of users being able to submit additional jobs while workloads are already being processed. In addition, while jobs are being queued and executed, the system is expected to be able to provide access to researchers and users as to the overall performance of their submitted jobs.

As most departmental owners of clusters empower end users with access to their own environment, it is of equal critical importance that end users can quickly view, edit, and modify on-the-fly jobs either executing or in queue. In the midst of this activity, a high priority job may be submitted that requires immediate processing. Our objective was to measure these factors and evaluate the results.

A third key performance indicator that revealed itself was the time taken to simply get in a position with each scheduler engine to actually perform the testing. As in other verticals of IT, time is often the most valuable currency. Where a system or set of tools is measurably easier to set up and execute work on, a direct relation of time arguably better spent performing work to ROI would likely be easy to establish.

Based on the results of the benchmarks performed to date, Platform LSF significantly out-performed the other platforms under evaluation. As noted previously, several runs were performed in each benchmark to establish consistent data as a basis. In addition, checks for errors in execution as well as accuracy of final results were evaluated in a side-by-side comparison from platform to platform.

In addition to performance dominance in the benchmark criteria, Platform LSF was consistently straightforward in the setup phase of each benchmark test, as well as being consistently reliable with regards to performance numbers generated in the tests themselves. Being consistent with performance numbers enables end users of the platform to accurately plan for job execution time and deadline forecasts upstream to end consumers of the data generated.

It should also be noted that while some of the platforms in evaluation offer commercial support, as well as additional software enhancements for users, the Platform LSF solution excels by offering a support infrastructure that spans all cycles of the procurement, installation, and on-going maintenance of the system. In addition, the software add-ons described in Appendix A can offer support and additional structure for nearly any research need, all available under one platform family.

Platform LSF should be at the top of any HPTC department's evaluation list for new and existing projects. The strong core competencies, as well as the enhanced features and user-friendly access combine to make Platform LSF a readily accessible infrastructure. Backed by strong commercial support and professional services, in addition to the strong performance numbers generated, Platform LSF is well poised to assist HPTC departments in their research initiatives, saving them time.

Appendix A – Additional Software Add-ons

IBM Platform LSF provides for additional, key features that users are able to leverage as additional software add-ons. A brief category description and product sheet follows:

Application Management	**IBM Platform Application Center** – Application-centric interface for both cluster administrators and users. This extension ties together site policies, addresses security concerns, and provides a reliable, self-documenting interface that results in cluster community ease of management and conformity.
Cluster Performance	**IBM Platform RTM** - Operational dashboard that ties together single or multi-cluster performance into a single pane of glass. Abilities include job profiling, cluster efficiency, and full visibility into how the cluster is performing at a glance and over time. Efficiency is improved with abilities including the shifting of workloads amongst clusters and the detection and use of idle capacity cluster-wide.
Analytics	**IBM Platform Analytics** - Service analytics engine that provides resource and service level analysis, in addition to correlation of usage to chargeback accounting, enabling organizations to schedule and account for COGs based on cluster usage.
License Management	**IBM Platform License Scheduler** - Provides not only valuable insight into licenses in use and management tools to more effectively deploy limited licenses to the cluster; this tool also provides the ability to configure sharing policies for licenses by way of a virtual license pool, and then schedule jobs with license constraints around job prioritization requirements and license ownership privileges.
Process Management	**IBM Platform Process Manager** - An automation engine that helps eliminate tedious, time consuming script management typically used to automate what can be highly complex, multiple step jobs. By using the Process Manager, a library of enforceable, best practices and step procedures can be automated and reused, saving valuable cluster programming time and potential user errors as a result of micro-management of highly complex, inflexible traditional scripting practices.

Session Management	**IBM Platform Session Scheduler** - Can help departments that have a high volume of short running, yet large volume jobs needing submission to be immediately dispatched, freeing up the platform queue for additional job batches and submissions. In addition, through the ability to pre-define resource requirements once for multiple jobs in the same session, time is saved both on the cluster and with end-users by eliminating tedious per-job resource programming considerations.
Cluster Management	**IBM Platform Dynamic Cluster** - Ties together both virtual and physical, separate cluster entities into a single, workload-aware computing cluster as a shared resource for users. Through the combination of tying together what typically is separate cluster installations and transforming idle resources into usable resource pools cluster-wide, the add-on can dramatically increase job submission rates and cluster utilization with better usage of available, yet typically physically different cluster hardware configurations, presenting to end users viable resource pools for job sessions that previously were not possible.
Data Management	**IBM Platform Data Manager for LSF** - Improves data throughput and minimizes wasted compute cycles to lower storage costs. Platform Data Manager automates the transfer of data used by application workloads running on Platform LSF clusters and the cloud. Frequently used data transferred between multiple data centers and the cloud can be stored in a smart, managed cache closer to compute resources.

Appendix B – Raw Test Results

Raw Test Results from Jobs Completed per Hour

Workload Manager	Run Sequence	Throughput
IBM Platform LSF	1	1666666
	2	1682242
	3	1666666
SLURM	1	253699
	2	255500
	3	258435
TORQUE	1	10997
	2	10969
	3	11000
Open Grid Scheduler	1	137404
	2	138089
	3	137825

Raw Test Results from Scheduler Query Time Tests

Workload Manager	Query Minimum	Query Average	Query Maximum
IBM Platform LSF	0.03	0.04	0.07
	0.02	0.04	0.06
	0.02	0.04	0.07
SLURM	0.14	0.64	9.49
	0.22	0.58	10.25
	0.21	0.3	2.22
TORQUE	0.01	0.04	0.1
	0.01	0.02	0.06
	0.01	0.02	0.07
Open Grid Scheduler	7.4	9.81	11.59
	7.92	9.88	11.38
	8.39	10.09	11.7

Raw Test Results from Scheduler Dispatch Tests

Workload Manager	Submission Minimum	Submission Average	Submission Maximum
IBM Platform LSF	0.01	0.02	0.02
	0.01	0.02	0.03
	0.01	0.02	0.03
SLURM	0.01	0.2	1.98
	0.01	0.06	1.34
	0.01	0.16	1.57
TORQUE	0.02	0.11	0.42
	0.02	0.08	0.53
	0.02	0.03	0.21
Open Grid Scheduler	0.02	6.55	8.53
	0.02	6.93	9.1
	0.02	6.97	8.77

Raw Test Results from High Priority Dispatch Time Tests

Workload Manager	Response Minimum	Response Average	Response Maximum
IBM Platform LSF	0	0.03	1
	0	0.07	1
	0	0.03	1
SLURM	0	10.43	47
	1	17.57	44
	1	16.23	37
TORQUE	4	84.93	185
	4	93.03	185
	1	97.37	189
Open Grid Scheduler	10	14.63	20
	11	15.16	22
	10	14.79	25

Raw Test Results from Scheduler Concurrent Queries Under Load Tests

Workload Manager	Concurrent Queries Success	Concurrent Queries Failure
IBM Platform LSF	15775	0
	15829	0
	15820	0
SLURM	347	0
	131	0
	128	0
TORQUE	3404	0
	3826	0
	4953	0
Open Grid Scheduler	3998	0
	3918	0
	3910	0

Appendix C – Workload Manager Versions Tested

The workload managers used, along with the version as of the publication of this paper, are presented below:

- IBM Platform LSF
  - Version 9.1.2
  - Source : lsf9.1.2_linux2.6-glibc2.3-x86_64.tar.Z
- SLURM
  - Version 2.6.5
  - Source : slurm-2.6.5.tar.bz2 (http://www.schedmd.com)
- TORQUE
  - Version 4.2.6.1
  - Source : torque-4.2.6.1.tar.gz (http://www.adaptivecomputing.com/support/download-center/torque-download/)
- Open Grid Scheduler
  - Version 2011.11p1
  - Source : GE2011.11p1.tar.gz (http://sourceforge.net/projects/gridscheduler/files/)