WHITEPAPER

High-Throughput Computing for HPC

Convergence of high-throughput computing (HTC)
with high-performance computing (HPC)

Adaptive COMPUTING

Table of contents

Introduction

As use of HPC clusters becomes more diversified, the industry is witnessing a convergence of high-throughput computing (HTC) with high-performance computing (HPC). Sectors once focused on HPC, such as electronic design automation (EDA), finance and insurance, chemistry, life sciences, oil and gas, manufacturing, and defense and intelligence, now need to optimize systems for both types of computing jobs.

While HPC workloads are compute- and data-intensive and can sometimes take several months to complete, HTC jobs have by nature extremely short runtimes, usually in the milliseconds to minutes range. Nearly all HTC jobs can be classified as "embarrassingly parallel," which means the workload can be divided up into multiple, autonomous pieces, each of which are capable of being independently executed. HTC jobs include some Monte Carlo simulations, molecular dynamics simulations, chip design, fraud detection, risk management, and many others.

Traditional HPC systems have focused on tackling scalability in the form of large batch jobs, or large computing environments—not necessarily in the form of speed and throughput. Consequently, requests for HTC support have plagued HPC administrators for quite some time, as their systems are not designed for its special needs. These pain points are accentuated by the ever-increasing call for elevating application performance to get more results with existing resources.

Existing software-based solutions are centered on managing larger workloads and increasing efficiency of HPC systems through scheduling and optimization. Depending on which policies govern prioritization, resource allocation and accounting, many schedulers address HPC problems relatively well.

There are plenty of high-throughput schedulers and solutions available in the marketplace. However, they are only for dedicated high-throughput systems, leaving HPC administrators with inadequate tools to respond to the growing job convergence.

The Bottleneck in High-Throughput Computing

The root of HTC on HPC's problems lies, in most cases, with the HPC scheduler's decision engine. HPC schedulers have made significant advancements in accelerating and automating complex decisions while taking into account many factors and policies. However, in the case of HTC jobs, a complex decision process is not necessarily needed for such simple computing jobs.

Unfortunately, it is not feasible to briefly disable complex scheduling in order to "manually" push these types of jobs through, as this would negate the optimization benefits of the scheduler. It would also wreak havoc on job queues, create a domino effect of inefficiencies, and result in failure to deliver SLAs to the organization.

Navigating around this decision process is a challenge that becomes exponentially more difficult when HTC jobs number in the hundreds of thousands and even millions. The scheduler cannot make decisions on all of those jobs. Even high-performing schedulers capable of pushing multiple jobs through in seconds would grind to a halt under such a load.

Need: A "Non-Scheduling" Solution

A scheduler is required to ensure a highly functioning, highly efficient HPC system, but it cannot make decisions on every job in a high-throughput scenario. Instead, the scheduler needs the added capability to make the decision on a batch of HTC jobs once, and place it on optimal resources at the right time.

Of those solutions available for HTC, some require vendor lock-in (generally undesirable), additional software from the vendor, or both. Few are battle-tested for commercial or high-production HPC systems, and may not fully integrate with existing schedulers.

Administrators are faced with the conundrum of needing a non-scheduling-based service to push millions of short-lived jobs through, without requiring individual job scheduling.

Solution: Push Tasks, Not Jobs

Enter Nitro from Adaptive Computing – a high-throughput scheduling solution for a traditional HPC system. Nitro is a highly efficient task launching software that operates independently of and integrates seamlessly with Moab Cloud for the HPC Suite, Adaptive Computing's workhorse job-scheduling software suite.

Instead of requiring individual job scheduling, Nitro enables high-speed throughput on short computing jobs by allowing the scheduler to incur the scheduling overhead only once for a large batch of jobs, effectively creating an Nitro "cell." Nitro then quickly and effectively executes the large batch of jobs within this pool without scheduling overhead. When pushing a high quantity of jobs through the system, Nitro works in unison with Moab Cloud to facilitate parallel job scheduling with larger, longer-running jobs.

Installing and using Nitro is very straightforward. It:

- features a simple user job submission
- can be used on existing clusters with no Moab Cloud upgrades or environment modifications
- works with other HPC schedulers, HPC clusters without a scheduler, and Hadoop environments

Nitro has the massive scale HTC needs, to the tune of millions, if not billions, of tasks. Nitro requires full nodes to operate, and can assign tasks to cells, or groups of nodes (also referred to as "workers").

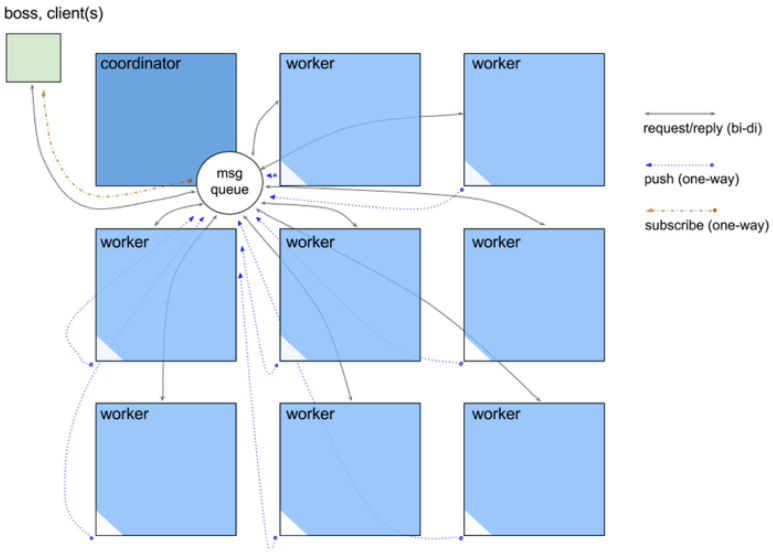
Optimal Use

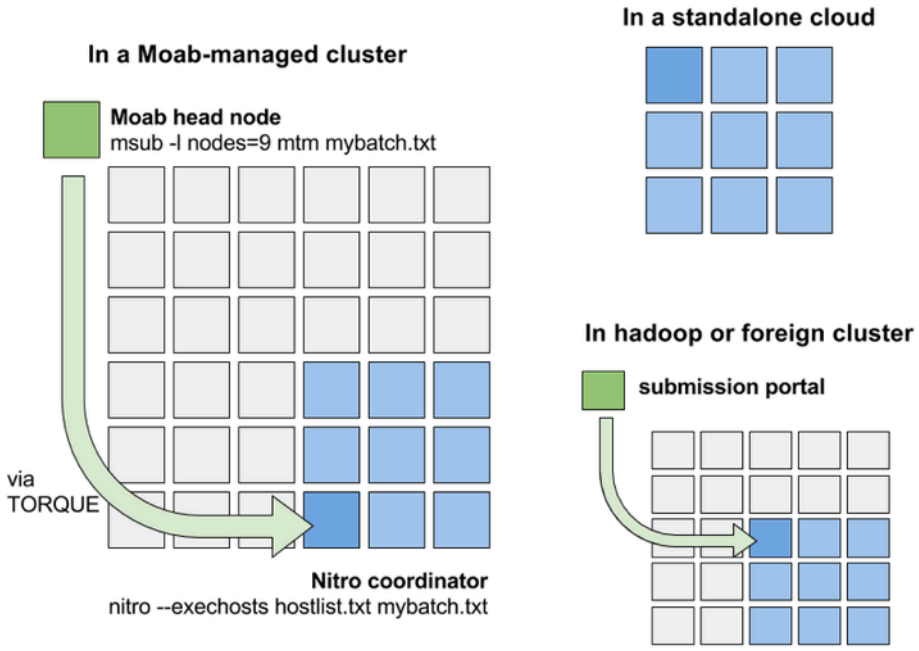
Nitro is highly valuable to administrators who want drastically improved throughput on jobs that run for milliseconds and request few resources. Naturally, some granularity in the management of individual tasks will be sacrificed, and all tasks in a batch must share resource and policy constraints. While not suitable for MPI, there are still multiple use cases where if even a fraction of the workload matches the high-throughput use cases covered by Nitro, it is well worth a look.

How It Works

Nitro operates by taking a subset of the nodes in an HPC cluster and turning them into a cell of nodes where tasks, or short-lived jobs, can be assigned. Nitro's coordinator, the head of a cell, receives a batch from the job scheduler, or a manual job submission, and delegates assignments from that batch out to each worker node in its cell. Each worker runs the assigned tasks, and reports back to the coordinator.

Each batch submission can build, activate, and tear down a new Nitro. This is seamless and requires no management by administrators or users. Individual Nitro cells can scale up to a few hundred nodes (enough to launch hundreds of thousands, or millions, of tasks per second). A given cluster can simultaneously host as many, or as few Nitro cells as needs dictate.





Hardware Requirements

Nitro is designed to work in a wide variety of environments and systems. Beyond HPC, Nitro can work on a standalone cloud, or even a Hadoop cluster. The hardware requirements for Nitro are very light: 500MB of RAM and two hardware threads. The more hardware threads given to Nitro, the faster Nitro can launch tasks. There are no limits on the batch size, and increasing the batch size does not increase hardware requirements.

Contact a solutions advisor by phone or email, or visit our web site today.

North America, Headquarters +1 (239) 330-6083
Provo, UT, USA Office +1 (801) 717-3700

Corporate Headquarters
704 Goodlette Road North
Naples, FL 34102

Email: info@adaptivecomputing.com
www.adaptivecomputing.com