White paper

In cooperation with Ericsson

Intel® delivers highly accurate simulations for planning and optimizing the Ericsson® cloud infrastructure solutions

Ericsson® Hyperscale Datacenter System 8000

Intel® CoFluent™ Technology for Big Data

Intel® CoFluent™ technology for Big Data (Intel® CoFluent™ technology) allows you to model and simulate clusters with a high degree of accuracy. In this study, we used the Ericsson® Hyperscale Datacenter System 8000, based on Intel® Rack Scale Design, to improve general performance for various cluster configurations. We then used Intel CoFluent technology to further model and simulate the performance of the different configurations. This helped us identify specific servers, storage, and network areas where an upgrade in components could be justified by a significant performance gain. The combination of Intel and Ericsson technologies can help you substantially increase the accuracy of pre-planning for cluster architecture, optimize specific component costs for your business needs, and minimize your total cost of ownership.

AUTHORS

Tomas Fredberg Ericsson
Chief Architect
Hyperscale HDS8000 Cloud Platform
Ericsson

Gen Xu
Software and Services Group
Intel Corporation

(AUTHORS CONTINUED ON NEXT PAGE)

ABSTRACT

Performance issues in the cloud can affect the performance of all applications that run on top of that cloud Infrastructure. In this paper, we demonstrate some of the capabilities of Intel® CoFluent™ technology for Big Data (Intel® CoFluent™ technology) that improve system throughput and performance.

In this study, we performed modeling and simulations on Ericsson® Hyperscale Datacenter System 8000 (Ericsson® HDS 8000)* for cloud platforms.[1] The Ericsson HDS 8000 is a software-defined infrastructure system, based on Intel® Rack Scale Design (Intel® RSD). The system allows for automated, software-based inventory of datacenter resources, and assembly of purpose-built servers from disaggregated pools of resources.

Intel CoFluent technology is a planning and optimization solution that identifies performance issues in hardware and software, such as in a cluster of servers. For example, using Intel CoFluent technology, we can examine the performance of different

AUTHORS — CONTINUED

Zhaojuan (Bianny) Bian
Software and Services Group
Intel Corporation

Illia Cremer
Software and Services Group
Intel Corporation

Mike Riess
Software and Services Group
Intel Corporation

hardware and software configurations. This helps design teams find the best solution for their business needs, based on performance versus component cost for cloud-based platforms and overall infrastructure. When paired with the Ericsson HDS 8000, Intel CoFluent technology allows design teams to easily adjust infrastructure configurations, to achieve maximum resource use and help minimize total cost of operations (TCO).

For this paper, we used Intel CoFluent technology to model, simulate, and compare an OpenStack Swift*[2]-based object storage system on the Ericsson HDS 8000. By using Intel CoFluent technology, we were able to identify the performance characteristics (including issues) of different object sizes. This allowed us to see how performance changed with different hardware configurations. Validation of the model shows a simulation accuracy that averages 95% or higher.[3]

Our work shows the value of using Intel CoFluent Technology to optimize cloud performance and build a more balanced system of compute, storage, and networking elements for varying workloads. Thanks to the Ericsson HDS 8000 and its ability to easily reconfigure network resources, we have been able to significantly improve the throughput of a Swift-based storage system over a software-optimized 10GbE infrastructure of large objects (>1MB). Specifically, our results demonstrate an excellent 2x throughput improvement in a 10Gb fabric configuration benefitting from CLOS network topology adopted in the Ericsson HDS 8000. We demonstrate an even greater throughput improvement of up to 4.8x in a 25Gb fabric configuration.[3]

Table of Contents

BACKGROUND

Ericsson Hyperscale Datacenter System 8000

Data centers are under severe pressure to meet the growing demands of applications for the cloud, big data, mobile devices, and social collaboration. Yet today's data centers are still built on traditional architectures where it can take days or even weeks to provision new services. These traditional data centers also typically run with poor utilization of server resources. This limits efficiency and flexibility, while at the same time, drives up costs. Fortunately, the evolution of cloud platforms is enabling greater efficiency in data centers via flexible, self-provisioning, standards-based interfaces.

The Ericsson® Hyperscale Datacenter System 8000 (Ericsson® HDS 8000) significantly improves data-center efficiency and speeds up service provisioning.

The Ericsson HDS 8000 is a logical architecture that is based on Intel® Rack Scale Design (Intel® RSD). Intel RSD is designed to help data centers handle always-on, ever-increasing demands. First, the logical architecture of Intel RSD disaggregates compute, storage, and network resources. This new rack design then introduces a new capability: Pooling resources so that assets can be used more efficiently.

Based on Intel RSD, the Ericsson HDS 8000 can simplify resource management and dynamically compose resources based on workload-specific demands. With the Ericsson HDS 8000, you can take the efficiency, flexibility, and agility of the cloud to the next level.

Intel® CoFluent™ technology for Big Data

Intel® CoFluent™ technology for Big Data (Intel® CoFluent™ technology) is a planning and optimization solution for big data clusters. With Intel CoFluent technology, you can use modeling and simulations to plan, predict, and optimize both hardware and software configurations. This helps you address common cluster design challenges that are becoming increasingly critical to solve. Such challenges include predicting system scalability, sizing the system, determining maximum hardware utilization, optimizing network behavior, and predicting cluster performance.

For software, Intel CoFluent technology simulates the software stack at functional levels. This includes the behavior of distributed file systems, operating systems (OS), and Oracle Java* virtual machines (JVM). Hardware activities derived from software operations are then dynamically mapped onto architecture models. Mapping is done for processors, memory, storage, and networking devices, according to workloads and performance.

During planning, Intel CoFluent technology helps you carefully evaluate various design choices by letting you swap out hardware components and/or change software elements. This helps you quickly evaluate the trade-offs between simulation speeds, accuracy, scalability, and complexity as you develop your cluster architecture. With rapid, effective evaluations, you can be more effective at predicting and optimizing both hardware and software before you begin provisioning systems.

Intel CoFluent simulations let you move away from trial-and-error planning or high level estimation-based planning. Instead, with Intel CoFluent technology, you can shift to high fidelity cluster simulation methodology.

This is an innovative simulation solution that makes it easier, faster, more accurate, and more efficient to perform capacity planning, performance evaluation, and optimization based on the trade-offs most appropriate for your business model. You can now more accurately plan according to your business needs, and identify optimal IT spending (instead of overspending) on big data clusters.

Levels of abstraction

Intel CoFluent technology for Big Data can abstract and simulate hardware and software at different levels, from simple abstractions to high fidelity descriptions of the cluster. For example, your Intel CoFluent technology model could be as detailed as a developer's behavior diagram. It could also be as simple as a group of black boxes that represent elementary mechanisms.

With a high level of abstraction you can model Kernel computing functions. With a low, detailed level of abstraction, you can simulate operations and algorithms related to sub-task or data splitting, scheduling, and tracking. At the low level of abstraction, you can also simulate activities that require memory usage, the network, or storage access that could become single points of bottleneck.

In this paper, we use a low, detailed level of abstraction to identify configuration-related performance issues.

Dynamic mapping of software and hardware

Big data applications typically run on cluster middleware. The cluster middleware divides a given application into sub-tasks. Each sub-task is dynamsically assigned to a node, according to your cluster topology, data location, real-time resource availability, and/or system payload.

After a simulation starts, activities derived from the software stack model are dynamically mapped onto the cluster's hardware components. The timing information is then extracted from the computation of the resource usages. This is sometimes called "late mapping." This type of mapping is required for simulation flexibility, as well as required by the cluster software stack, since sub-tasks are usually allocated dynamically to hardware resources.

Layered simulation architecture

It's important to understand your software architecture in order to get the most out of your simulations and models. With Intel CoFluent technology, the software behaviors and hardware architecture of the cluster are loosely coupled. This gives you the flexibility to change cluster architecture without having to modify the software behavior model, and vice versa.

To achieve this loose coupling, as well as enable the mapping between software model and hardware architecture, Intel CoFluent technology uses a layered simulation architecture (see Figure 1):

- Layer 1: Software stack layer. This is the top layer, where the cluster software stack behavior is modeled.
- Layer 2: System topology layer. This layer defines the cluster's hardware components, such as the processor, network, and storage. The software roles of each physical node are also assigned in this layer. For example, the Apache Hadoop* File System (HDFS) data node and the Map/Reduce task tracker are assigned in this layer.
- Layer 3: Resource monitoring and performance library layer. This layer tracks the hardware resource usage, and produces the timing information required by the simulation. This layer includes resources such as CPU/memory, storage, switches, routers, cluster fabric, and so on.
- Layer 4: Simulation engine. The lowest layer is the simulation engine for SystemC-based discrete events. This is a low-overhead engine that enables fast simulations and good scalability.
- Vertical. A vertical module interacts with the top three layers of the simulation architecture. This vertical module performs the dynamic mapping of hardware and software elements. In turn, this drives the realistic simulations of the performance of your cluster's design.

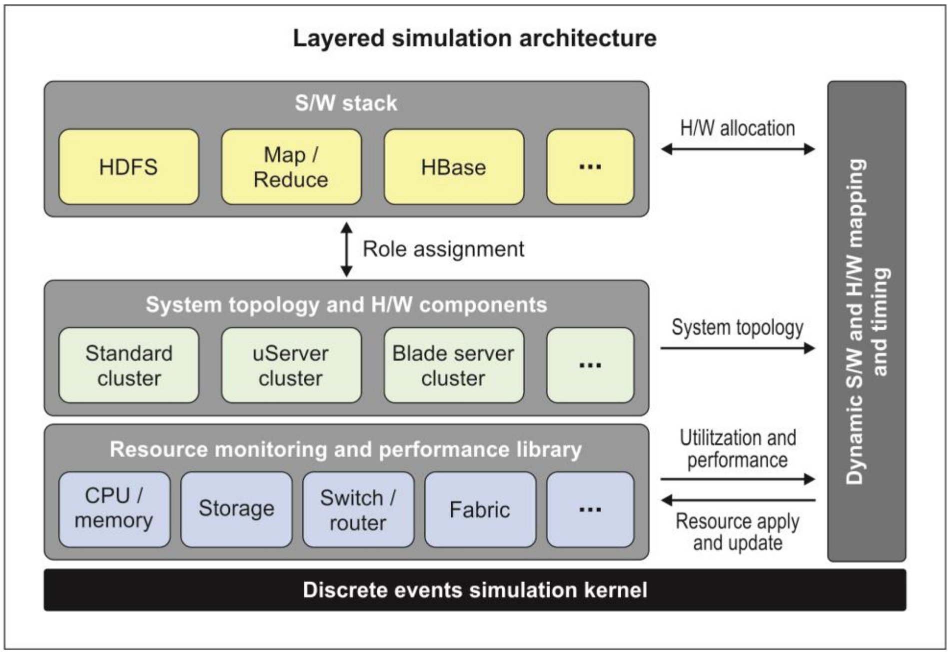

Figure 1. Layered approach to simulations

Table 1. Initial hardware and software stack configurations

Hardware configuration	
Network switch	Intel® Ethernet Multi-host Controller FM10420
Server nodes: 1 Proxy server 2 Storage servers 1 Client server	Intel® Xeon® processor E5-2695 v3 64GB RAM 1 mSATA SSD attached as the operating system disk 20 SSDs attached to each storage server
Network interface card	10/25Gb fabric
Software configuration	
Operation system	Canonical Ubuntu 15.04*
Openstack Swift* version	Openstack Swift Liberty*

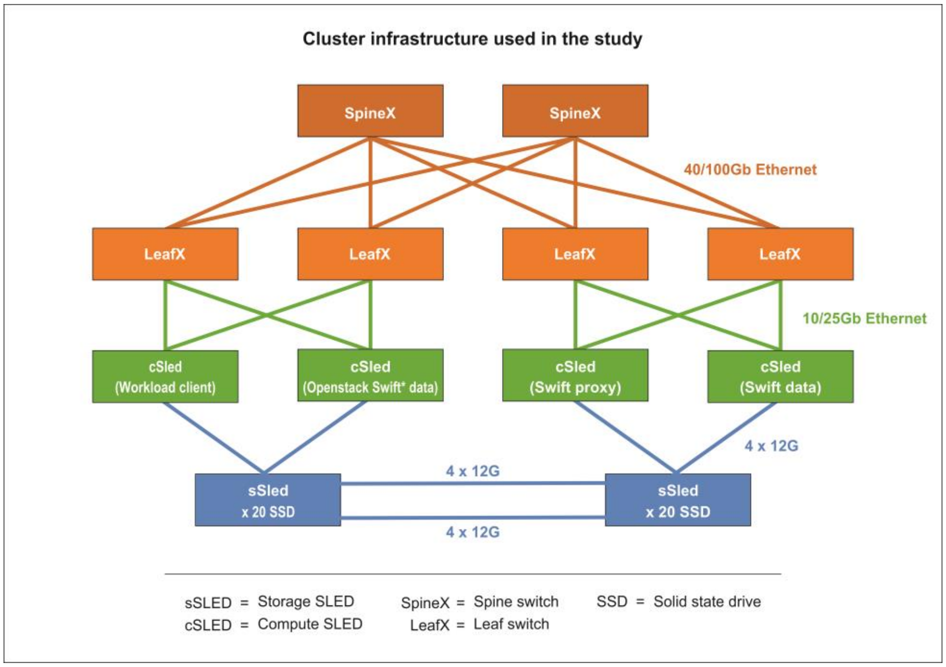

Figure 2. The Ericsson® Hyperscale Datacenter System 8000 configuration used in our simulation.

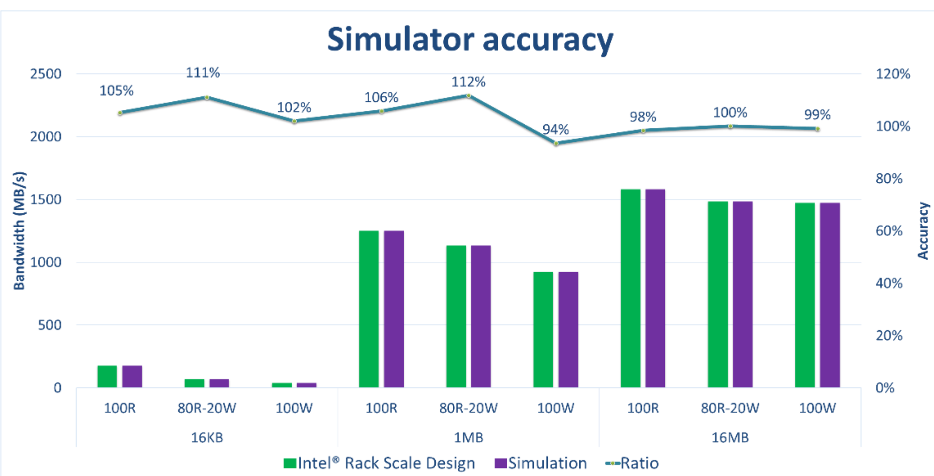

Figure 3. Simulator accuracy[3]

EXPERIMENT SETUP

Baseline configurations

Table 1 (previous page) lists the target Openstack Swift* cluster hardware and software stack used for our baseline configuration. The topology of the cluster is shown in Figure 2.

Benchmark

The data presented in this paper is based on the COSBench benchmark.[4] COSBench is a representative and comprehensive benchmark that evaluates the performance of cloud object-storage services. To study performance for various scenarios, we chose three kinds of operations and three typical object sizes.

Object operations include putting, getting, and mix operations. The object sizes of 16Kb, 1MB, and 16MB represent tiny objects, medium-sized objects, and large objects, respectively. You can think of the three object sizes as representing three typical scenarios: text storage, image storage, and music storage.

SIMULATION RESULTS

Simulation accuracy

Before simulating each configuration using Intel CoFluent technology, we used COSBench to validate the accuracy of the simulations on a range of physical hardware and software configurations. We present the results here, to illustrate the high degree of accuracy of the simulator. For all scenarios examined, average simulation accuracy was 95% or higher, as shown in Figure 3.[3]

Once we were confident in the high degree of accuracy of the simulations, we were ready to use Swift workloads and Intel CoFluent technology to demonstrate the capacity of the Ericsson HDS 8000.

Consider trade-offs when selecting optimal hardware components for your cluster

Before deploying the Swift cluster, we had to consider trade-offs that might be required to meet both storage capacity demands, as well as the requirements of service-level agreements and/or service-level objectives (SLA/SLO). Moreover, we also needed to be prepared for cluster growth and any issues that might come up in trying to meet future demands.

Using the simulator to help set the cluster target, we wanted to specify a cost-effective system that could still scale as needed to provide sufficient capacity and performance to meet project goals. Here, we used the simulator to help build an effective cluster by selecting the appropriate storage, network, and computer hardware resources.

Identify performance characteristics of different types of storage

Swift offers cloud storage in which many types of data (as objects) can be stored and retrieved. In any cluster, the ability to quickly access non-sequential data is a key performance consideration.

For tiny objects (16KB), the objects are randomly written to or read from storage devices. We know that hard disk drives (HDDs) have a slow random-access speed, so updating HDDs to solid state drives (SSDs) should dramatically improve the cluster's total throughput. Moreover, the sequential speed of SSDs is several times higher than that of HDDs, so using SSDs should, again, increase throughput.

Therefore, the first step in improving existing storage performance was to simulate replacing the conventional HDDs with SSDs in the cluster. That upgrade should allow us to identify specific advantages of the dramatically higher read and write IOPS of SSDs. We could then identify the workloads where the higher cost of the SSDs is justified by the improvement in performance.

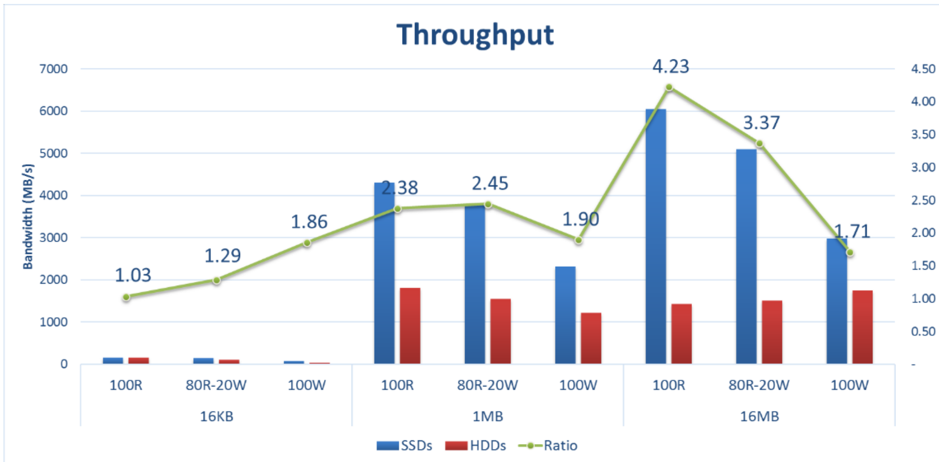

Figure 4. Optimizing storage[3]

Simulation results for the upgrade are shown in Figure 4. As shown in the figure, you can see that the benefit of the higher access speed of SATA SSDs depends on the size of the object being stored and retrieved. Depending on the size of the object, the upgrade to SSDs delivers various gains. For tiny objects, throughput ranges from 1.03x to 1.86x the performance of conventional HDDs.[3] For large objects, the performance gain is as much as 4.23x.[3]

The data indicate that SSDs can be useful for objects of all sizes, in clusters that handle almost every type of workload, and are especially useful for workloads that handle medium and large objects. This includes active document/content/financial archiving, web application backend, database storage, scientific data serving, and music/video streaming.

Identify throughput characteristics of different networks

The distributed nature and replication policy of Swift storage makes network I/O another vital aspect of overall cluster performance. So the next area to examine for potential upgrades is the network.

Network topology and speed are the two most important factors of network throughput. We know the 10Gb Ethernet (10 GbE) and fat tree topology are the most common networks, and have been widely used in data centers. We also know that the test hardware for our Ericsson HDS 8000 includes higher performing network devices and CLOS topology that can allow for greater throughput.

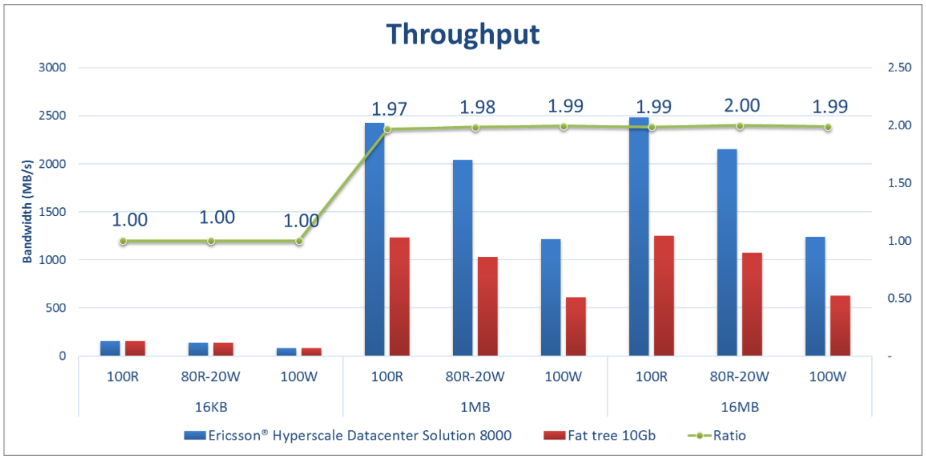

Figure 5. Throughput of Ericsson Hyperscale Datacenter System 8000 compared to a fat tree cluster with the same configuration[3]

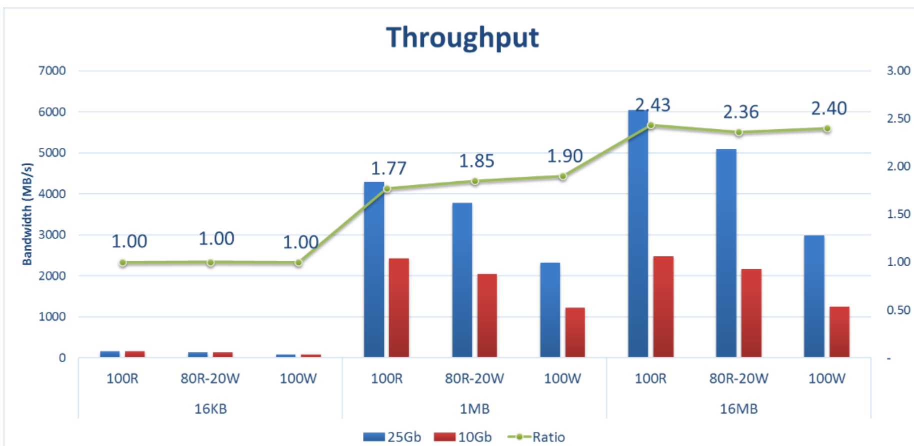

Figure 6. Optimizing network performance[3]

Figure 7. Performance improvement of the Ericsson® Hyperscale Datacenter System 8000 compared to a fat tree cluster with the same configuration[3]

Our simulations show that the throughput of the test cluster in our Ericsson HDS 8000 already has almost 2x the throughput of a 10GbE fat tree cluster that has the same server configuration (see Figure 5).[3] In the network configurations we explored, the Ericsson HDS 8000 configuration shows the greatest increase in throughput for medium-sized and large objects.

When we upgraded the network from 10GbE to 25GbE (see Figure 6), throughput increased even further, with up to 2.43x the previous performance.[3] Performance increased by a negligible amount for the smallest objects, but showed a significant gain for medium-sized and large object sizes.[3] (The benefits of different network configurations in real systems have been measured and correlated very well to the simulation.)

Ultimately, the performance improvement of the Ericsson HDS 8000 results in a total of about 4.8x, as compared to the original 10GbE fat tree cluster (see Figure 7).[3]

The performance gains for medium-sized and large objects is particularly crucial for workloads such as image serving and data archiving. This can help us optimize component cost versus throughput performance in clusters that handle those types of workloads. For example, for these workloads, we should use the Ericsson HDS 8000, and upgrade the network fabric for those specific clusters to 25GB.

Identify performance characteristics of different compute resources

With our previous upgrades to storage and network components, our I/O system became much faster than our initial baseline configuration. Specifically, we saw significantly reduced or eliminated CPU wait times. However, we felt that I/O efficiency could be improved further.

Tiny objects — and even medium-sized objects — can be CPU intensive. We simulated upgrading CPUs from Intel® Xeon® E5-2695 v3 2.3GHz processors to Intel® Xeon® E5-2697 v3 2.6GHz processors. In doing so, we saw improved and consistent scaling for both small and medium-sized objects (see Figure 8).

Processing of small and medium-sized objects is important for workloads such as text storage, image serving, and music streaming. In our cloud cluster, the study's results show that, to optimize compute efficiency versus component cost we should consider upgrading the specific servers that will handle those kinds of workloads.

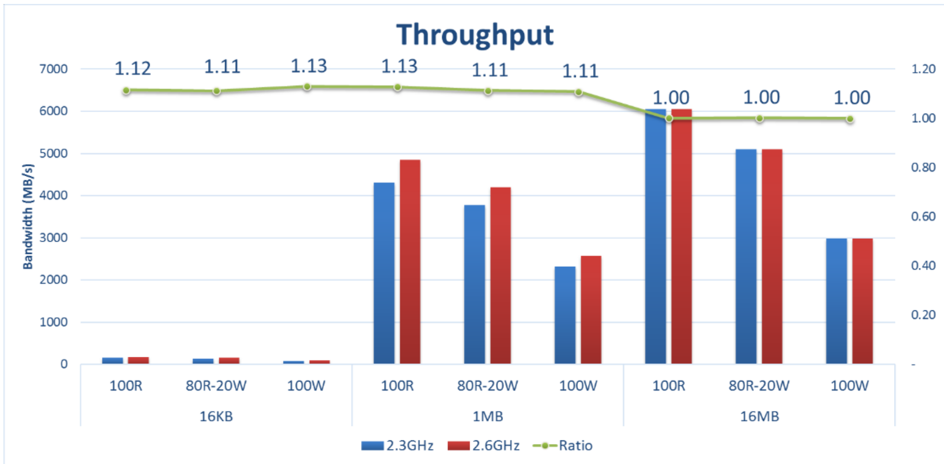

Figure 8. Optimizing compute efficiency[3]

CONCLUSION

It would be easy to say, upgrade everything to get better performance, but that's neither practical nor cost-effective. Efficient cluster optimization requires a strong understanding of both the software tasks being executed, and the hardware being used for each type of task.

Traditionally, this kind of optimization has revolved around the operators' experience and estimations, and has proven to be effective with that expertise. However, software and hardware interactions in today's clusters are typically very intricate. This makes optimization difficult even for highly experienced operators.

In addition, as systems scale and increase in complexity, and correspond to even greater financial investments, the need for precise and quantified optimization and planning is more important than ever.

The simulation and modeling capabilities of Intel CoFluent technology for Big Data provide significant and specific information about the best way to make performance improvements. They also provide a timely, scalable, more accurate, and more cost-aware solution for optimizing complex systems.

Experimental results for Swift workloads demonstrate the high degree of accuracy of these simulations: Average errors are below 5% across the scaling of more than 10 software and hardware configurations.[3]

With Intel CoFluent technology you can now more accurately model complex systems. You can do so even where software and hardware elements are abstract representations of system behavior and performance characteristics. These simulations can be used to effectively identify system issues and recommend balanced system configurations according to different usage scenarios (object sizes, read/write ratios, and so on).

In this study, we have shown that, thanks to the configurable high-speed network fabric of the Ericsson HDS 8000, Intel CoFluent technology can help significantly improve throughput. Specifically, we have shown how to improve throughput of a Swift-based storage system over a standard 10GbE fat tree infrastructure of large objects (>1MB).

Our results demonstrate an excellent 2x throughput improvement in a 10Gb fabric configuration benefitting from CLOS network topology adopted in the Ericsson HDS 8000, and an even greater throughput improvement of up to 4.8x in a 25Gb fabric configuration.[3]

Even more, we have used highly accurate simulations to show exactly where the performance improvements occur. With Intel CoFluent technology, you can be more confident early in the design cycle in choosing the best combination of components for your business needs. With Intel CoFluent technology, you can better optimize critical performance parameters while minimizing development costs and TCO.

Learn more about accurate modeling and simulation technologies

For information about Intel CoFluent technology,
including Intel CoFluent technology for Big Data, visit **http://cofluent.intel.com**

For more information about the Ericsson HDS 8000 cloud solution, visit
https://www.ericsson.com/hyperscale/cloud-infrastructure/hyperscale-datacenter-system

[1] For information about the Ericsson Hyperscale Datacenter System 8000 (Ericsson HDS 8000), refer to the Ericsson HDS 8000 solution brief, "Modernize Your Datacenter with Hyperscale Architecture," on the Ericsson Web site.

[2] Openstack Swift is a distributed object storage system. For more information, see https://wiki.openstack.org/wiki/Swift

[3] Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

[4] COSBench is a benchmark tool for cloud object storage system. For more information, visit https://github.com/intel-cloud/cosbench

Printed in USA XXXX/XXX/XXX/XX/XX Please Recycle XXXXXX-001US