White Paper

Highly accurate simulations of big-data clusters for system planning and optimization

Intel® CoFluent™ Technology for Big Data

Intel® Rack Scale Design

Using Intel® CoFluent™ Technology for Big Data to model and simulate clusters can significantly improve the accuracy of optimizations for performance versus component cost. Using Intel® Rack Scale Design can further improve performance for various cluster configurations. The combination of technologies can substantially increase the accuracy of pre-planning cluster architecture, help optimize component costs for your business needs, and help minimize total cost of ownership.

Authors

Gen Xu
Software and Services Group
Intel Corporation

Zhaojuan (Bianny) Bian
Software and Services Group
Intel Corporation

Illia Cremer
Software and Services Group
Intel Corporation

Joe Gruher
Data Center Group/Rack Scale Design
Intel Corporation

Mike Riess
Software and Services Group
Intel Corporation

ABSTRACT

Performance issues in the storage system can affect the performance of all applications that run on top of that storage system. 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. Testing was performed with Intel® Rack Scale Design hardware. This hardware 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 different hardware and software configurations to find the best solution for performance versus component cost for a Big Data cluster. When paired with Intel Rack Scale Design hardware, configurations can be easily adjusted to achieve maximum resource utilization and help minimize total cost of operations.

For this paper, we used Intel CoFluent technology for Big Data to model, simulate, and compare an OpenStack Swift*[1]-based object storage system on an Intel Rack Scale Design. 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.[2]

Our work shows the value of using Intel CoFluent Technology to optimize cluster performance versus cost, and build a more balanced system of compute, storage, and

networking components to better handle the different types of workloads. Thanks to Intel Rack Scale Design and its ability to easily configure network resources, we have been able to significantly improve the throughput of a Swift-based storage system over a fixed 10GbE infrastructure of large objects (>1MB). Specifically, our results demonstrate an excellent 3x throughput improvement in a 25Gb fabric configuration, and an even greater throughput improvement of up to 5x in a 50Gb fabric configuration.[2]

TABLE OF CONTENTS

BACKGROUND

Intel® Rack Scale Design

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 weeks to provision new services. These traditional data centers also typically run with poor utilization of server resources. This limits efficiency and flexibility and, at the same time, drives up costs.

Fortunately, the evolution of cloud platforms is enabling greater efficiency in the data center via flexible, self-provisioning, standards-based interfaces.

Intel® Rack Scale Design significantly improves data-center efficiency and rapid service provisioning.

Intel Rack Scale Design is a logical architecture designed to help data centers handle always-on, ever-increasing demands. First, Intel Rack Scale Design disaggregates compute, storage, and network resources. This new rack design then introduces a new capability: Pooling resources for more efficient utilization of assets.

With Intel Rack Scale Design, you can simplify resource management and dynamically compose resources based on workload-specific demands. Intel Rack Scale Design helps you 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 plan, predict, and optimize 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, OS, and Oracle Java* virtual machines (JVM). Hardware activities derived from software operations are then dynamically mapped onto architecture models 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 swapping out hardware components or changing software elements. You can quickly evaluate the trade-offs between simulation speeds, accuracy, scalability, and complexity as you develop your cluster architecture. This helps you be more effective at predicting and optimizing both hardware and software before you begin provisioning systems.

Intel CoFluent technology simulation capabilities let you move away from trial-

and-error planning and high level estimation-based planning. With Intel CoFluent technology, you can shift to high fidelity cluster simulation methodology.

This innovative simulation solution facilitates more accurate and more efficient 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, with Intel CoFluent technology, your model can be as detailed as a developer's behavior diagram, or as simple as a group of black boxes that represent elementary mechanisms.

> ***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 dynamically 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.

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 describe a simulation with a low, detailed level of abstraction in order to identify configuration-related performance issues.

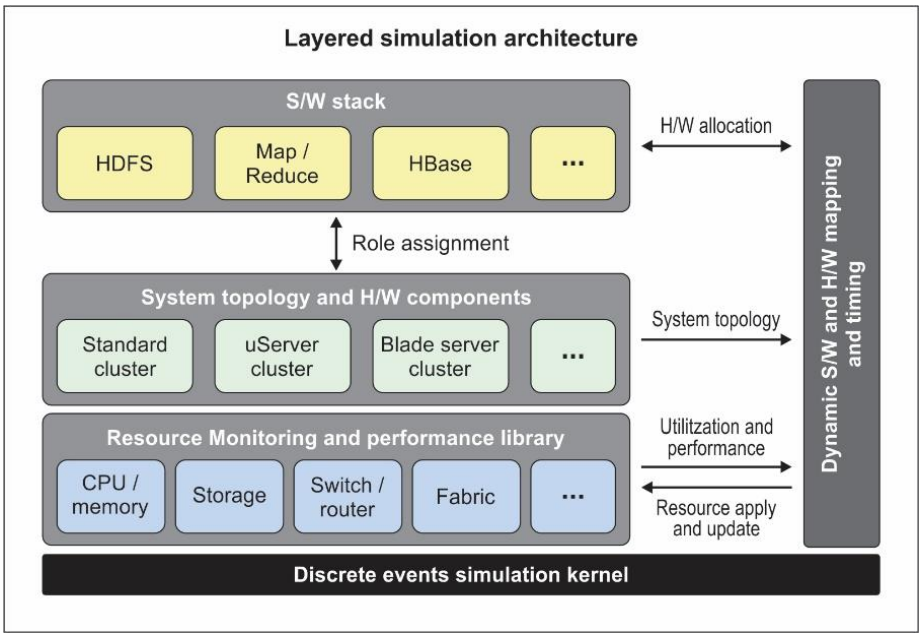

Figure 1. Layered approach to simulations

Layered simulation architecture

It is 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, above):

- 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 (for example, the Hadoop distributed file system, or HDFS; data node; and the Map/Reduce task tracker) are also 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 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 to perform the dynamic mapping of hardware and software elements. This drives the realistic simulations of the performance of your cluster's design.

EXPERIMENT SETUP

Baseline configurations

Table 1 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 (below).

Table 1. Hardware and software stack configurations

Hardware configuration	
Network switch	Intel® Ethernet Multi-host Controller FM10420
Server nodes: 2 Proxy servers 4 Storage servers 2 Client servers	Intel® Xeon® processor E5-2695 v3 64GB RAM 1 mSATA SSD attached as the OS disk 14 HDDs attached to each storage server
Network interface card	25Gb fabric
Software configuration	
Operation system	Canonical Ubuntu 15.04*
Openstack Swift* version	Openstack Swift Liberty*

Benchmarks

The data presented in this paper is based on the COSBench benchmark.[3] 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 16Kb, 1MB, and 16MB represent tiny objects, medium-sized objects, and large objects, respectively. The three object sizes can be thought of as representing three typical scenarios: test storage, image storage, and music storage.

After simulating each configuration using Intel CoFluent technology, we tested the accuracy of the simulations on actual configurations using COSBench. We present the results using these simple parameters to illustrate the high degree of accuracy of the simulator.

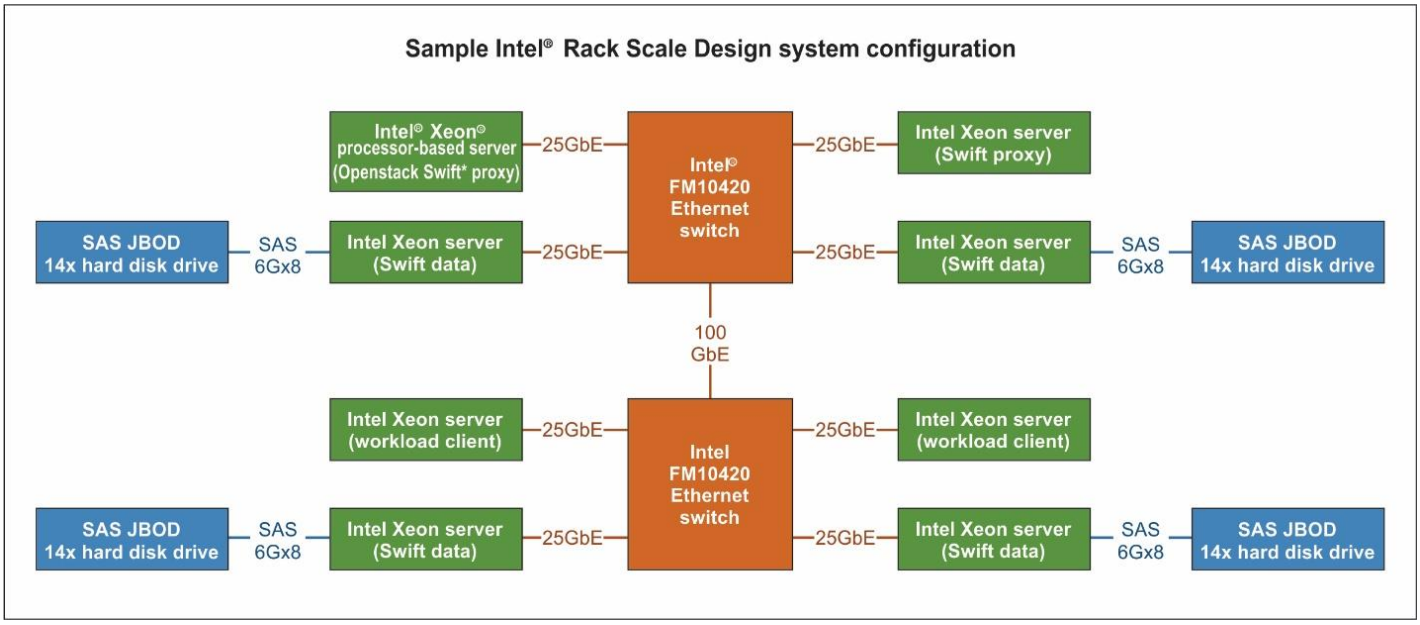

Figure 2. Intel® Rack Scale Design system configuration used in our testing. (In the figure, SAS refers to Serial-attached SCSi; and JBOD refers to "just a bunch of disks.")

SIMULATION RESULTS

Identify and resolve performance issues

Performance issues in the storage system can affect the performance of all applications that run on top of the storage system.

In this paper, we use OpenStack Swift, which is a distributed storage system. Because the storage system is distributed, configuration issues in the OS, software stack, or hardware stack can dramatically affect overall performance in the cluster. Intel CoFluent technology simulations can quickly identify these kinds of configuration-related performance issues.

In our study, we first use Intel CoFluent technology to verify that the cluster's hardware and software components are functioning at the best possible out-of-the-box performance level.

To verify out-of-the-box performance, after Swift was installed, we used COSBench to measure the system throughput.

When we compared that data with our simulation numbers, we found that the throughput of small object writes in the physical system was much lower than the simulation numbers. We also observed that there was a low performance phase between the 53rd and 113th seconds during execution. This indicated that the OS was not properly configured.

After we updated the OS configuration, the measurement numbers of the physical system matched the simulation numbers. This indicated that we had established an appropriate hardware and software configuration.

Verify simulation accuracy

The next step after defining the baseline configuration was to validate the accuracy of the simulation. Here, we used empirical data to validate the simulator's results. For all scenarios examined, average simulation accuracy was 95% or higher, as shown in Figure 4.[2] Once we were confident in the high degree of accuracy of the simulations, we were ready to use Intel CoFluent technology to help deploy and optimize a Swift cluster.

Consider trade-offs when selecting optimal hardware components

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

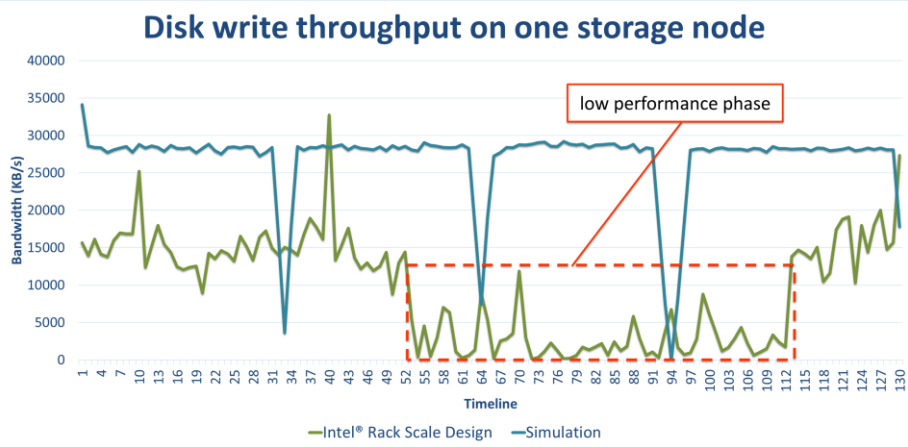

Figure 3. Disk write throughput on one storage node[2]

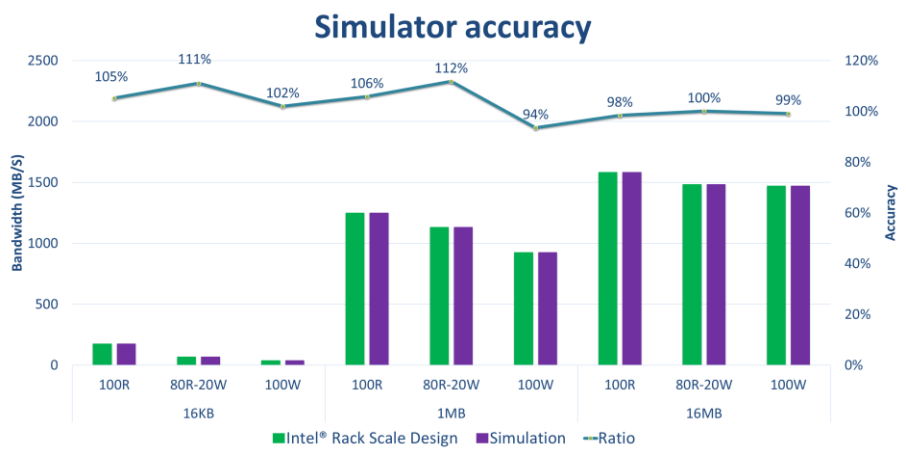

Figure 4. Simulator accuracy[2]

Using the simulator to help set the cluster target, we know we can architect a cost-effective system that can still scale as needed to provide sufficient capacity and performance. 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 (such as objects) can be stored and retrieved. The ability to quickly access non-sequential data is a key performance consideration in any cluster.

In our experiment, 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. Swift also consumes network bandwidth heavily due to its replication characteristics, so SSDs should again, be a better choice in this cluster.

Therefore, the first step in improving existing storage performance is to replace the conventional HDDs with SSDs in our simulation. That upgrade should allow us to identify specific advantages of the dramatically higher random read and write IOPS of SSDs.

Simulation results for the upgrade are shown in Figure 5. When we verified simulation results on actual hardware, the simulation results were very close to the hardware measurements, with an average error of below 5%.[2]

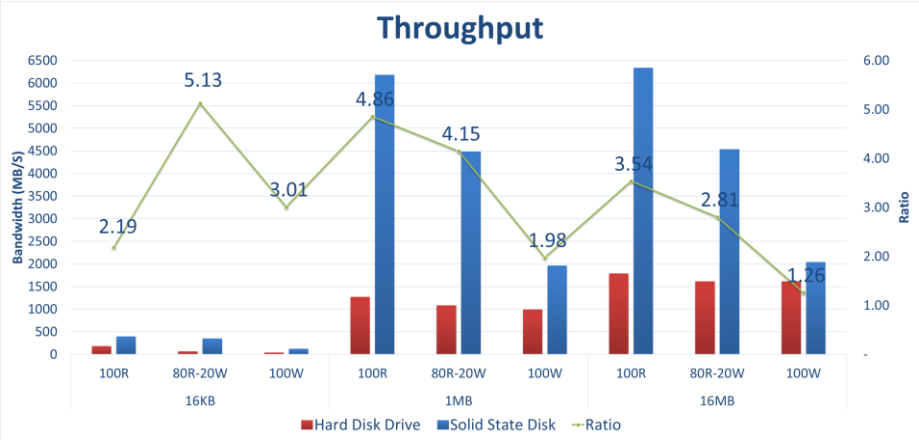

Figure 5. Optimizing storage[2]

As shown in Figure 5 (above), you can see that the benefit of the higher random access speed of SSDs depends on the size of the object being stored and retrieved. Depending on the size of the object, the upgrade to SSDs delivers a throughput of up to 5.13x the performance of conventional HDDs.[2]

The data indicate that SSDs can be particularly useful in clusters that handle workloads with a high number of medium-sized (about 1MB) and larger (about 16 MB) objects. Such workloads include video streaming, database storage, scientific data storage, active document/content/financial archiving, and Web application storage.

Identify throughput characteristics of different networks

The distributed nature and 3-copy replication of Swift storage makes network I/O another vital aspect of overall cluster performance.

Because of this, the next area to upgrade is the network. We know that the 10Gb Ethernet is the most common type of network and has been widely used in data centers. We also know that our Intel Rack Scale Design test hardware includes higher performing network devices, and can allow for greater throughput.

Our Intel CoFluent technology simulations show that the throughput of the test cluster in our Intel Rack Scale Design has almost 3x the throughput of a 10GbE cluster that has the same server configuration (see Figure 6, next page).[2] The greatest increase in throughput occurred for medium-sized and large objects.

When we upgraded the network from 25Gb to 50Gb (see Figure 7, next page), throughput increased even further, with up to 1.77x the previous performance.[2]

Performance increased by a negligible amount for the smallest objects, but again, increased significantly for medium-sized and large object sizes.[2] (The benefits of different network configurations in real systems have been measured and correlate very well to the simulation.)

The performance gains for medium-sized and large objects is particularly crucial for workloads such as video streaming, database storage, and Web application storage.

The simulations show that, to optimize component cost versus throughput performance in clusters that handle those types of workloads, we should use Intel Rack Scale design and upgrade the network fabric for those clusters to 50GB.

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 image serving and music streaming. In our big-data cluster, to optimize compute efficiency versus component cost for that type of workload, we should consider upgrading the servers handling those workloads to Intel Xeon E5-2697 v3 2.6GHz processor-based servers.

CONCLUSION

It would be easy to say, upgrade everything to get better performance. That is 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.

However, software and hardware interactions in today's clusters are typically very intricate, which makes optimization difficult even for highly experienced operators.

In addition, as systems scale and increase in complexity, and correspond

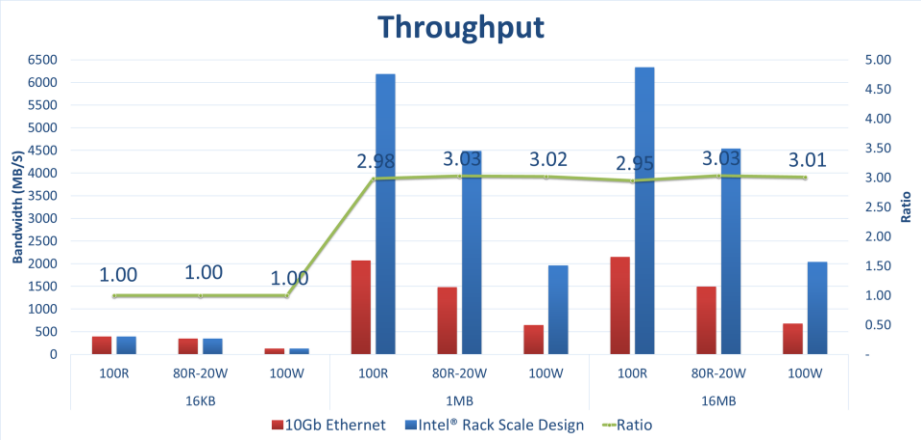

Figure 6. Throughput of Intel® Rack Scale Architecture as compared to 10GbE[2]

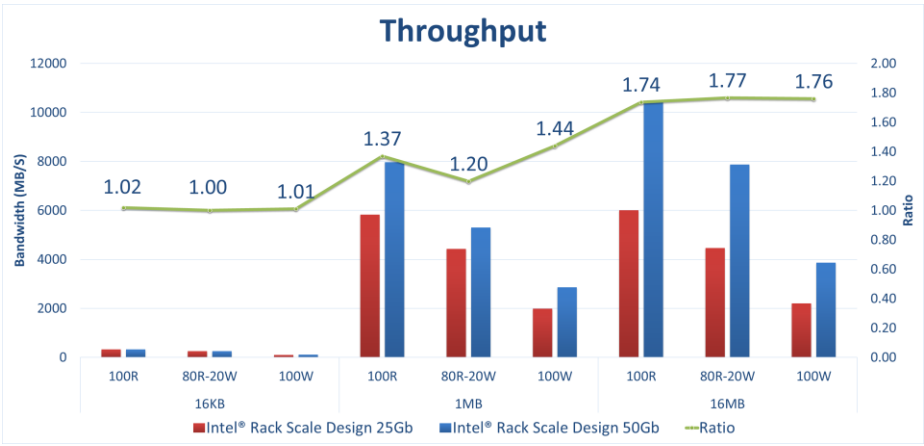

Figure 7. Optimizing network performance[2]

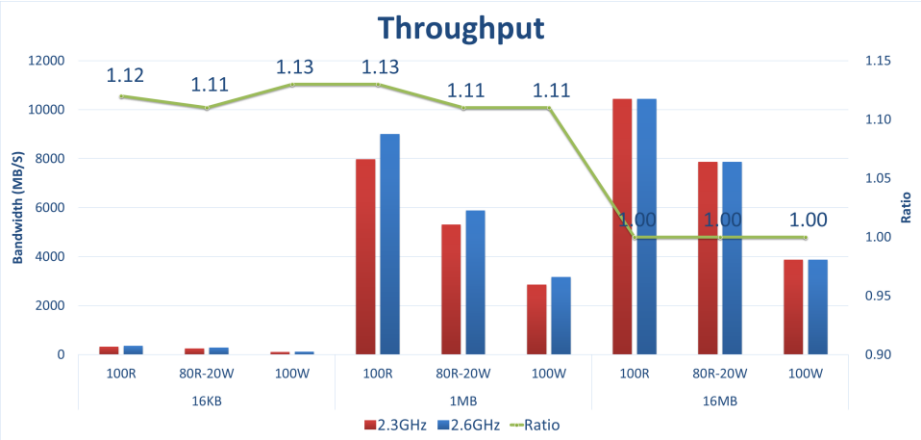

Figure 8. Optimizing compute efficiency[2]

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 performance improvements. They also provide a timely, scalable, more accurate, and more cost-aware solution for complex system optimization.

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.[2]

With Intel CoFluent technology you can now more accurately model complex systems even where software and hardware elements are abstract representations that capture 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).

Here, we have shown that, thanks to the configurable, high-speed network fabric of Intel Rack Scale Design, Intel CoFluent technology can help significantly improve the throughput of a Swift-based storage system over a standard 10GbE infrastructure of large objects (>1MB).

Specifically, our results demonstrate an excellent 3x throughput improvement in a 25Gb fabric configuration, and an even greater throughput improvement of up to 5x in a 50Gb fabric configuration.[2]

Even more specifically, we have used highly accurate simulations to show exactly where the performance improvements occur.

With Intel CoFluent technology for Big Data, you can now be more confident early in the design cycle in accurately choosing the best combination of components for your business needs. With Intel CoFluent technology, you can optimize critical performance parameters while minimizing development costs, component costs, and total cost of operations.

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 Intel Rack Scale Design, visit http://intel.com/intelrsd

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

[2] 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.

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