UNIVERSITÀ DI PISA

IMDT FOR CONTAINERS

Technical Report

ABSTRACT

Intel® Optane™ solid state technology offers a persistent media with capacity and pricing typical of disk drives but latency that is not so far from the memory. The Intel® Optane™ P4800X drive come with Intel Memory Drive Technology (IMDT), a software that allows extending the memory of a system using the solid-state drive, moving the classic tiering from storage tiering to memory tiering. This shift has several potential benefits: larger memory available to programs while keeping costs under control. On the other hand, latency of the Intel® Optane™ media is superb when compared to other solid-state technologies but still slower than memory.

In this report we extended the memory of a DellEMC® server with Intel® Optane™ P4800x and Intel Memory Drive Technology to measure how a memory intensive HPC workload is affected by the software memory tiering.

Our tests show that by tripling the memory (from 192GB to 698GB of RAM) the performance loss is acceptable in the range of 30% with respect to the memory only workload. The final figure is a worst-case scenario since we used a memory intensive workload, a mesh preprocessing from fluid dynamics simulation, typical of HPC.

The benchmark has been conducted using Docker containers to maximize the jobs throughput per unit of time given the large quantity of available memory.

Antonio Cisternino, Maurizio Davini

TR-01-18

Containers?

Why use containers?

Containers provide a solution to two key problems in IT, especially in large-scale deployments:

1. **Multi-tenancy:** Modern server computers are very powerful and can run many workloads concurrently. But how can one execute many workloads on the same machine, when each of the workloads potentially requires a different operating environment? For example, a software which runs perfectly while using a specific library or component (e.g. a specific Java Virtual Machine, or specific version of Perl) may misbehave if run on an environment where those components are different. And it's not just different software stack differences that can cause problems, it is also the environment: network topologies, storage layouts, security policies, etc.
2. **Workload migration:** The DevOps paradigm adds operational agility and financial efficiencies; releases of software improvements are created and deployed continuously, and internal and external IT resources are used interchangeably. But how can one ensure that software can be deployed and run efficiently and reliably when moved from one computing environment to another, e.g. from a developer's laptop to a QA environment, from staging to production, or from local IT facility to a public cloud and even concurrently on both?

What are containers and how do containers solve this problem?

From a technical perspective, a container is a chrooted environment which also implements process isolation, and depending on container product, possibly network isolation, and some filesystem services.

From a functional perspective, a container typically implements an entire runtime environment: an application, plus all its dependencies, libraries and other needed binaries, as well as configuration files and any other files needed to run it, bundled into one package. By containerizing the application platform and its dependencies, the differences stemming from different OS distributions and underlying infrastructure are abstracted away. Further, through process isolation, a container is assigned its own namespace from the host kernel, which is important for multi-tenancy as a program in the container can't access kernel memory or use more resources such as CPUs or RAM than it was allowed.

Containers also isolate other aspects of the environment. For example, isolation of network stacks can allow two applications/processes to listen on port 443 (assuming the system administrator correctly handles routing at the host level).

An important DevOps aspect is that containers have facilities, that allow respawning "clean" environments, so while a regular installation, even if chrooted, is read/write (changes are persistent), container can easily be configured start from a clean filesystem "baseline" each time you launch the container (e.g. using aufs).

The aforementioned are the key traits that make containers so popular for cloud environments.

Containers in HPC

Many of the benefits of using containers could be achieved by using virtual machine (VM) technologies that became very popular in IT from early this millennium. But VMs come with a cost, even the free and open-source ones: the cost of performance overhead and opaqueness of hardware resources. Due to these shortcomings, VMs never gained popularity in the HPC domain (High Performance Computing), and the promised benefits of "cloud" such as agility and improved resource utilization were not tapped into by the HPC community, as those users prefer to squeeze every ounce of performance out of their IT investments, even if it comes at the cost of complexity of operations.

With containers, many of the roadblocks to adopting cloud operation methodologies in HPC have been removed. HTC (High Throughput Computing) is a special case of HPC for which the benefits of cloud operation, especially with containers, are immediately apparent.

The issue with memory

As mentioned earlier, the servers are very powerful these days, with a COTS (commodity, off-the-shelf) dual-socket system easily reaching more than 30 CPU cores; thus – one can host dozens of containers, concurrently running dozens of workloads, on a single server. Alas, the computing power of the system is not the only resource that the processes use – memory and I/O are also needed and are sometimes in short supply. Consider an HPC workload with a varying memory usage profile ranging from 40GB to 50GB during runtime. If one were to run 32 of those concurrently, they would need to have more than 1600GB of DRAM to be sure that the machine does not run out of memory at any point during the execution of the workload throughput. Provisioning this amount of memory into a COTS server is either impossible or very expensive.

Intel® Optane™ with Intel Memory Drive Technology

In 2017, Intel has announced availability of products based on their new storage class memory (SCM) media named 3D Cross Point. One of the products using this new media is the Intel ® Optane^{TM} SSD DC P4800x series (Non-Volatile Memory PCI Express Solid State Drive), which combines this novel media with an I/O controller that connects directly to the computer's PCI bus; at the same time, Intel also announced the availability of Intel® Memory Drive Technology (IMDT), which transparently transforms Intel ® Optane^{TM} SSDs into DRAM expansion.

With these new technologies, the memory-size and -cost barriers have been removed, and we can now expand a COTS server to 6TB without buying any specialized expensive DRAM. It is only reasonable that we would try to see if these product from Intel also resolve the limitations imposed on efficient Cloud HPC deployments.

The benchmark

Methodology

To evaluate the viability of the Intel® Optane™ SSD DC P4800X Series with Intel® Memory Drive Technology (IMDT) approach to HPC Cloud, we selected to measure the increase in overall throughput we could gain by expanding the memory of a Dell R740xd system far beyond the installed DRAM capacity. Since we did not have very large memory in our possession, we scaled-back the test: we used a system with 192GB DRAM, and expanded it to 698GB total memory using two Optane SSDs with IMDT.

Test system and environment

Machine in DRAM-only configuration:

- 192GB RAM
- 2 X Intel(R) Xeon(R) Gold 6130 CPU @ 2.10GHz

Machine in memory expansion configuration:

- 192GB RAM
- 2 X Intel(R) Xeon(R) Gold 6130 CPU @ 2.10GHz
- 2 X Intel® Optane™ SSD DC P4800X Drives 375GB with IMDT

- Total amount of memory available to operating system and applications: 698GB

Software stack:

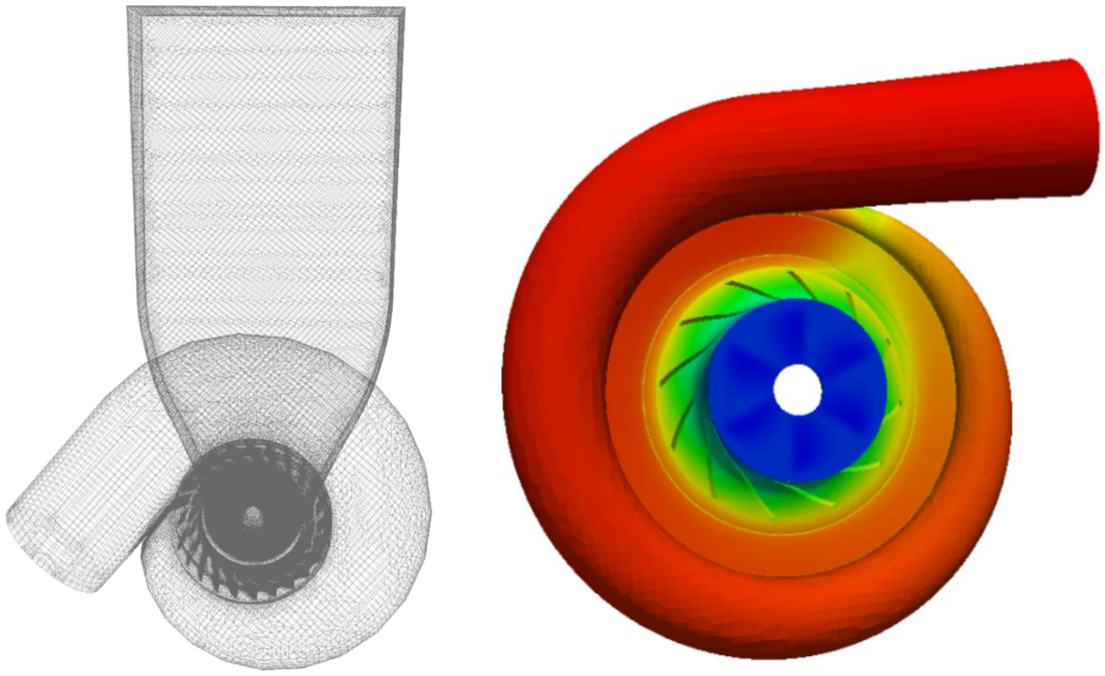
OS version: CentOS Linux release 7.4.1708

Kernel version: 3.10.0-693.5.2.el7.x86_64

IMDT version: 8.2.1455.74

Docker version: 17.10.0-ce

OpenFoam docker image: `openfoam/openfoam5-paraview54`



Test description

The selected workload represents a real-life case where a mesh is decomposed for fluid dynamics simulation workload.

In fluid dynamics simulation it is common to start from a mesh and chisel it to refine the space partition used by the simulation; the more a mesh is refined, the more accurate is the simulation. This step is crucial in a simulation since mesh generation and decomposition prepares the input for a parallel execution step, usually to be executed on a cluster, and may require significant amount of memory. It is also a processing step that is highly memory intensive, constituting a *worst case* type of workload for IMDT technology. In the past, the memory of the head node of an HPC cluster was enough to run a single decomposition, but the increased capacity of servers nowadays allow to run multiple decompositions simultaneously in the form of containers.

In our tests each container executed a mesh refinement and then decomposition step. We measured the completion time with a progressively high number of concurrent containers running the same workload. We performed the measurements using memory only (up to exhausting the system memory) and memory extended with Intel® Optane™ SSD DC P4800X drives with Intel® Memory Drive Technology

*Other names and brands may be claimed as the property of others.

The peak memory usage (per container) measured at 54GB, which means we could only run 5 containers concurrently before the system ran out of memory.

With the Intel® Optane™ SSD DC P4800X drives with Intel® Memory Drive Technology we had a total of 698 GB memory, and were able to run 18 containers concurrently.

Looking at completion time it is evident how difficult it is to use all the available cores due to constraints on the available memory. The use of IMDT slightly slows down the computation due to the use of solid state media, though the slowdown has been always contained allowing a better use of the available resources on the server and an overall increase in throughput of mesh generation jobs.

In order to have a better figure of the overall output we report the throughput we measured as number of similar jobs per hour. In this way it is possible to appreciate the contribution of the memory extension in allowing better utilization of the server, even when the workload is close to exhaust the available resources.

The following table shows the results of the experiment:

Results

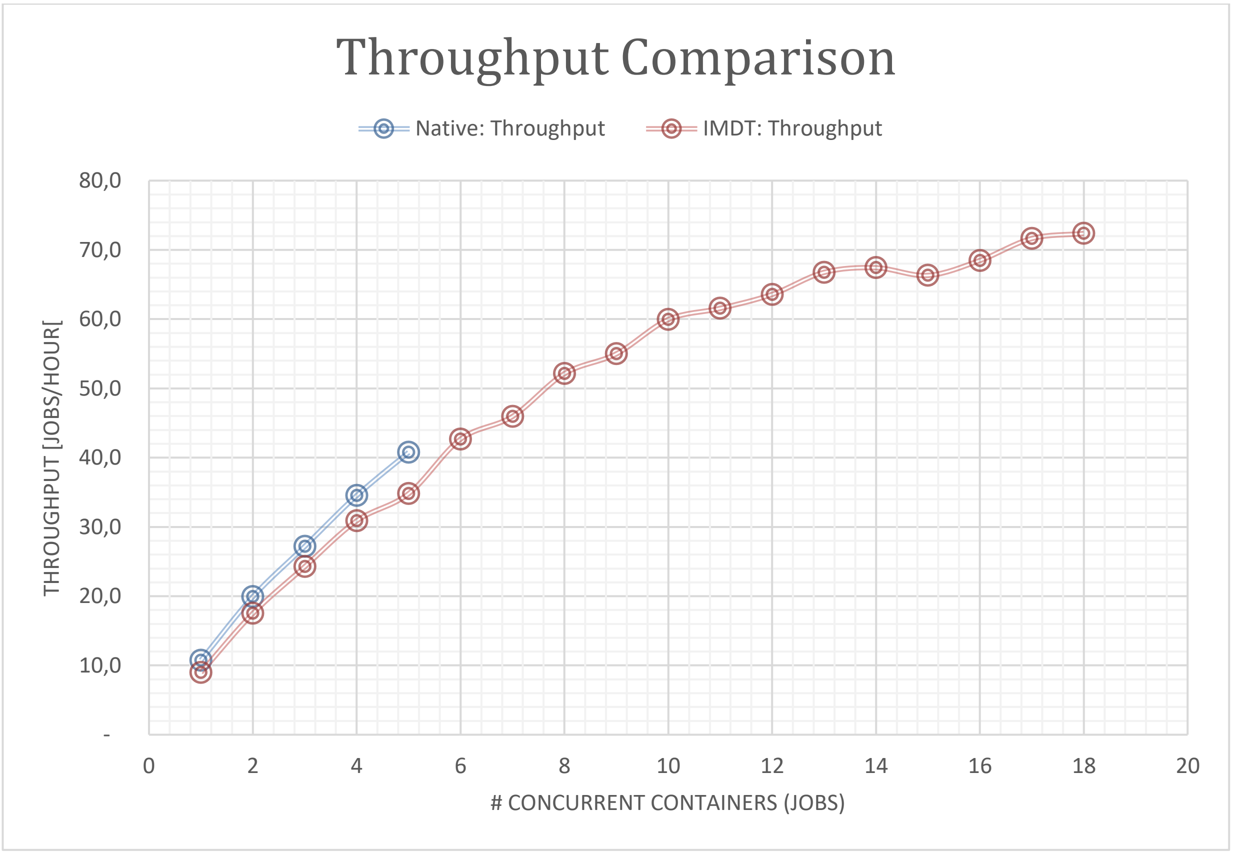
Native machine, DRAM-only:

# of Containers	Average Job Runtime [sec]	Throughput [Jobs/Hour]
1	335.2	10.7
2	361.3	19.9
3	397.1	27.2
4	417.0	34.5
5	441.6	40.8

IMDT:

# of Containers	Average Job Runtime [sec]	Throughput [Jobs/Hour]
1	400.37	9.0
2	411.08	17.5
3	445.08	24.3
4	466.11	30.9
5	517.19	34.8
6	506.10	42.7
7	548.39	46.0
8	552.39	52.1
9	589.22	55.0
10	600.50	60.0

11	643.30	61.6
12	679.49	63.6
13	701.02	66.8
14	747.54	67.4
15	814.22	66.3
16	841.17	68.5
17	854.20	71.6
18	895.45	72.4



Conclusion

By adding Intel® Optane™ SSD DC P4800X SSDs with Intel® Memory Drive Technology we are able to achieve 70% more throughput, fully maximining the available system cores, without the need to add an additional system. We are able to effectively capitalize on the benefits of cloud operations in an HPC environment by increasing infrastructure utilization.

*Other names and brands may be claimed as the property of others.