Using Azure CycleCloud and Batch for Scalable HPC Workloads

Colby Ford

For computational scientists at academic institutions and research organizations, the usage of on-premises high-performance computing (HPC) environments is paramount to scientific success. In fields such as physics, bioinformatics, and engineering, HPC environments are used to perform huge-scale computational workloads across multiple nodes of a large cluster.

In Microsoft Azure, there are two services that mimic on-prem HPC setups and provide an even more dynamically scalable environment for whatever workload you want to throw at it: Azure CycleCloud and Azure Batch. HPC is also called "Big Compute" and, with Azure, we can push the limits on how "big" we can go!

Why use HPC in the Cloud?

Traditional HPC environments require a large, air-conditioned room with a team of IT professionals that manage a few million dollars worth of equipment. Also, for anyone who has ever worked on a local HPC cluster, you know that scheduling is a nightmare. It's common for submitted jobs to take days to even start because of inefficient cluster management or too many people clogging up the machines.

Some benefits for using a cloud-hosted HPC environment:

- Lessens the need of a huge IT team to be HPC pros to manage the cluster(s)
- No need to buy millions of dollars in depreciating computing equipment
- Can dynamically scale up or down, reducing compute costs by only paying for what you use
- Using multiple HPC clusters can reduce user frustration for slow/shared job queues

EXAMPLE USE CASES:

Genetic Sequence Analysis	3D Image Rendering	Financial Risk Modeling
Optical Character Recognition	Multi-node Deep Learning	Fluid Dynamics
Finite Element Analysis	Media Transcoding	Semiconductor Design

Azure CycleCloud is one of the HPC services that provides an enterprise-friendly way to deploy cluster environments in Azure. This tool is quite flexible in that you can bring your own scheduler (like Slurm or OpenPBS) and define the types of virtual machines (VMs), scalesets, and networking that you want to support. In addition, CycleCloud also supports auto-scaling plugins that can interact with the supported schedulers and deployment templates.

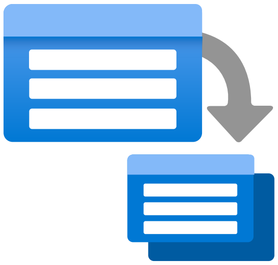
slurm workload manager	OpenPBS

There are multiple deployment templates that have been curated for all kinds of workloads, from 3D rendering in Blender to containerized cluster applications in Docker. You can see the full list here.

Not ready to fully give up on your on-prem HPC cluster just yet? No worries. CycleCloud can manage a hybrid setup with both on-prem and Azure-based machines.

To learn more about Azure CycleCloud, click here.

Azure Batch



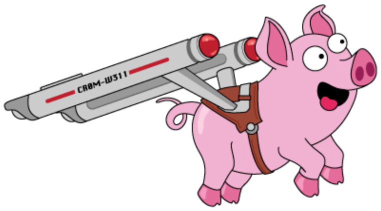
Azure Batch is another HPC service that allows for executing batch jobs efficiently. In fact, this service is quite similar to Azure CycleCloud, except that Batch provides a *Scheduler-as-a-Service*. So, there's no need to be a pro at Slurm or PBS, as you can just utilize the Batch Management API for managing Pools or Jobs.

Azure Batch provides the ability to run tightly coupled workloads where applications that are running need to communicate with one another, rather than running independently. This is a perfect use case for cloud-based technologies, as these machines can dynamically scale without loosing the ability to relay messages to each other.

Also, there is no specific charge for using Batch. You only pay for the underlying virtual machines while they're running.

To learn more about Azure Batch, click here.

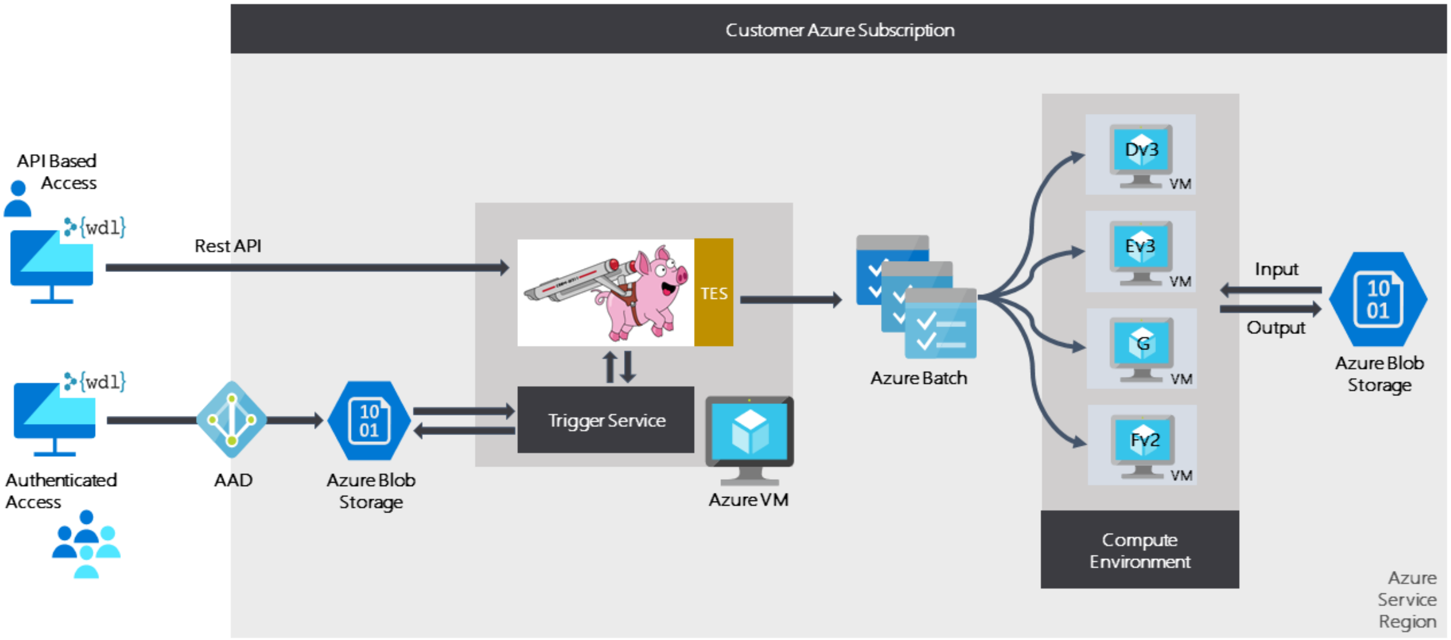
Sample Architecture: *Cromwell on Azure Batch*



As an example, let's look at running a large parallel processing workflow for Genomics on Azure Batch with Cromwell.

Cromwell is a workflow management tool by the Broad Institute for automating genome analyses at-scale. This software can be run on a local machine or in an HPC environment (and in Azure Batch).

Once deployed on Azure Batch (which only takes a couple steps), a user can interact with the Cromwell system via AAD authentication or via an API. This then triggers the service to create a job and distribute that workload across multiple virtual machines (VMs) in the compute environment. *See the architecture diagram below.*



Also, note that all of the VMs share a common data repository. In this case, it's Azure Blob Storage, but it could also be a data lake or a database.

Important consideration - make sure to pick the appropriate VM types for your workload:

- GPU workloads (like deep learning): N-series
- High memory needs (like in bioinformatics): M-series
- Super speed Xeon CPUs (for faster local parallelism): F-series

- Check out other VM types here

To learn more about Cromwell on Azure Batch, click here.

But, What About Azure Databricks (and Spark)?

For most distributable software seen in scientific computing, the logic to distribute workloads is controlled by the Message Passing Interface (MPI). MPI controls the tasks that are sent to nodes of a cluster and how to collect the responses when the tasks are finished. Though Spark uses MPI deep under the hood, Spark-enabled applications utilize a higher level engine to better distribute data and tasks to achieve "embarrassingly parallel" distribution.

While you can run MPI-based software on a Spark cluster, you'll end up paying for the Databricks Unit (DBU) charge on Azure Databricks, without actually getting any benefit from the Spark system itself.

My recommendation would be to use Azure Databricks for large-scale data and machine learning workloads where you can utilize the Spark engine (SparkSQL or Spark MLlib). Databricks is also useful with workloads that require more interactive development, like in a notebook IDE. Then, for other workloads that are distributed solely through MPI, use a cloud-based HPC option like Azure CycleCloud or Azure Batch for automation and scale.

To learn more about Azure Databricks, click here.

Want to Get Started?

Contact us today and get help from our team of data platform architects, bioinformaticians, and AI pros to help scale your workloads in the Azure Cloud.