**Domain: Develop Azure Infrastructure as a Service compute solutions**

Contents

[Topic: Implement batch jobs by using Azure Batch Services 2](#_Toc23508120)

[Sub Topic: manage batch jobs by using Batch Service API 2](#_Toc23508121)

[What is Azure Batch? 2](#_Toc23508122)

[Run parallel workloads 2](#_Toc23508123)

[Azure Batch Procession Architecture: 3](#_Toc23508124)

[Azure Batch Core Features: 3](#_Toc23508125)

[Azure Batch Limits and quotas 6](#_Toc23508126)

[Resource quotas 6](#_Toc23508127)

[Cores quotas in user subscription mode 7](#_Toc23508128)

# Topic: Implement batch jobs by using Azure Batch Services

## Sub Topic: manage batch jobs by using Batch Service API

# What is Azure Batch?

Use Azure Batch to run large-scale parallel and high-performance computing (HPC) batch jobs efficiently in Azure. Azure Batch creates and manages a pool of compute nodes (virtual machines), installs the applications you want to run, and schedules jobs to run on the nodes. There is no cluster or job scheduler software to install, manage, or scale. Instead, you use Batch APIs and tools, command-line scripts, or the Azure portal to configure, manage, and monitor your jobs.

Developers can use Batch as a platform service to build SaaS applications or client apps where large-scale execution is required. For example, build a service with Batch to run a Monte Carlo risk simulation for a financial services company, or a service to process many images.

There is no additional charge for using Batch. You only pay for the underlying resources consumed, such as the virtual machines, storage, and networking.

## Run parallel workloads

Batch works well with intrinsically parallel (also known as "embarrassingly parallel") workloads. Intrinsically parallel workloads are those where the applications can run independently, and each instance completes part of the work. When the applications are executing, they might access some common data, but they do not communicate with other instances of the application. Intrinsically parallel workloads can therefore run at a large scale, determined by the amount of compute resources available to run applications simultaneously.

Some examples of intrinsically parallel workloads you can bring to Batch:

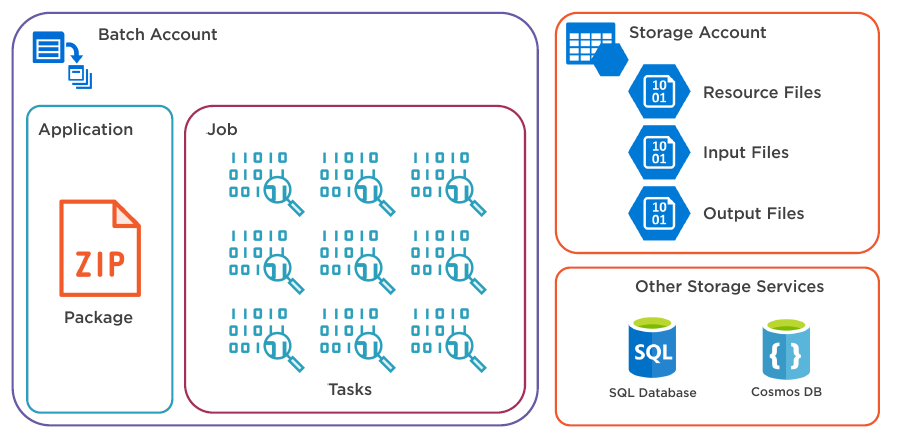
|  |  |  |
| --- | --- | --- |
| Financial risk modeling using Monte Carlo simulations | VFX and 3D image rendering | Image analysis and processing |
| Media transcoding | Genetic sequence analysis | Optical character recognition (OCR) |
| Data ingestion, processing, and ETL operations | Software test execution |  |

You can also use Batch to run tightly coupled workloads; these are workloads where the applications you run need to communicate with each other, as opposed to run independently. Tightly coupled applications normally use the Message Passing Interface (MPI) API. You can run your tightly coupled workloads with Batch using Microsoft MPI or Intel MPI. Improve application performance with specialized HPC and GPU-optimized VM sizes.

Some examples of tightly coupled workloads: Finite element analysis, Fluid dynamics and Multi-node AI training

Many tightly coupled jobs can be run in parallel using Batch. For example, perform multiple simulations of a liquid flowing through a pipe with varying pipe widths.

## Azure Batch Procession Architecture:



## Azure Batch Core Features:



**Resource Management:**

* Node Management (Cloud Service Worker roles, VM’s 3rd Party applications)
  + Cloud Service Worker Roles
    - Stateless VM
    - Platform as a service PaaS
    - Limited OS
    - Limited customization and temporary storage
  + Virtual Machines
    - Infrastructure as a service (IaaS)
    - Durable stateful storage
    - Many OS with full access to OS and specification options
  + 3Rd party applications
    - Rendering applications (Maya, 3ds Max, Arnold, Chaos Group V-Ray)
    - Hourly billing.
* Auto-Scaling
  + Set the number of nodes in a pool
  + Defined in a formula based on
    - CPU percentage, Memory
    - Number of tasks (Active, Running, Pending, Failed)
    - Retrieve task and resource data, adjust pool size based on that data.
    - Comprised of up to 100 statements, uses service-defined and user defined variables
    - Example of Auto scale formula: Auto scaling based on queue length

$MaxNodes=20;

$CurrentActiveTasks= $ActiveTasks.GetSample(1);

$CurrentRunningTasks=$RunningTasks.GetSample(1);

$TotalTasks=$CurrentActiveTasks + $CurrentRunningTasks;

$Nodes=min($TotalTasks,$MaxNodes);

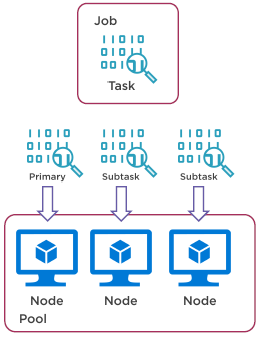
$TargetDedicated=$Nodes

**Common Service defined variables**

|  |  |
| --- | --- |
| **Read-only** | **Read-Write** |
| $CPUPercent | $TargetDedicatedNodes |
| $MemoryBytes | $TargetLowPriorityNodes |
| $ActiveTasks | $NodeDeallocationOption  requeue, terminate, taskcompletion, retaineddata |
| $RunningTasks |  |
| $PendingTasks |  |
| $SucceededTasks |  |
| $FailedTasks |  |

* Low Priority Nodes
  + Reduced resource cost processes task when capacity is available.
  + Useful when
    - Cost must be minimized
    - Job processing is not time critical
    - Maximizing resources with a fixed budget
  + Avoid when
    - Job processing is time critical
* Applications
  + Zip package containing executables
  + Deployed to nodes when required
  + Can be owned by Pools and Tasks
  + Pools- Deployed to all nodes in a pool
  + Tasks – Deployed to nodes running the task

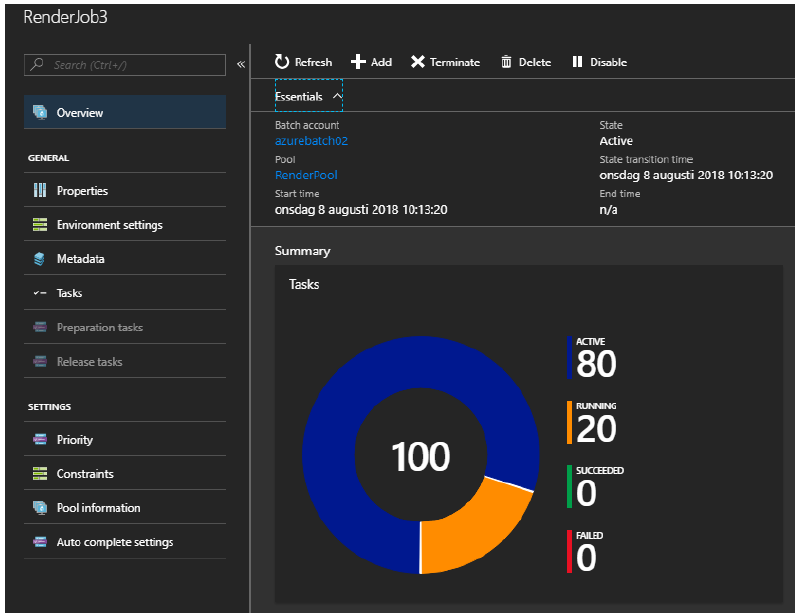
**Process Management (**Job and task management, Task dependencies, multi-instance tasks)

* Job and task management
  + Job & task scheduling
  + Job Prioritization
  + Re-try tasks on failure
  + Maximum task duration
* Task dependencies
  + Tasks can have dependencies on other tasks like input to T2 Depends on output of T1
  + Explicitly defined on task creation, used for scheduling task appropriately
* Multi Instance Tasks
  + Run tasks on multiple compute nodes
  + Allows for message passing interface (MPI
  + Inter-node communication is required

**Resource and Process Monitoring: (Azure portal monitoring, batch labs monitoring, application insights)**

Metrics available through API (REST, .NET, Azure CLI, Azure PowerShell)

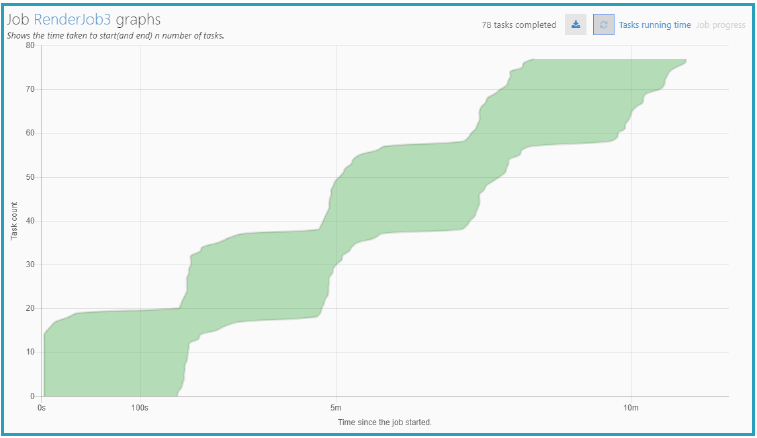
* Azure portal monitoring:

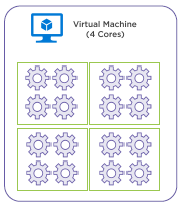


* Batch Explorer Monitoring:

**Note: Batch explorer can be downloaded and installed to local machine.**

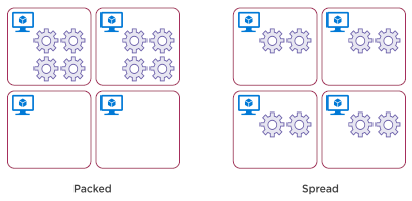




* Application insights:
  + Monitoring service used in many Azure projects it is like cloud watch in AWS
  + Node performance counters
  + Instrumentation in code

# Optimizing Job Processing

* Max tasks per node
  + Optimize resource usage
  + Upto 4 X Core count of node i.e. Concurrent tasks per compute node = 4 x number of node cores (This cannot be changed)
* Task Scheduling policy
  + Spreading = distribute across nodes
  + Packing = minimize node utilization



# Azure Batch Limits and quotas

As with other Azure services, there are limits on certain resources associated with the Batch service. Many of these limits are default quotas applied by Azure at the subscription or account level.

If you plan to run production workloads in Batch, you may need to increase one or more of the quotas above the default. If you want to raise a quota, you can open an online customer support request at no charge.

## Resource quotas

A quota is a credit limit, not a capacity guarantee. If you have large-scale capacity needs, please contact Azure support.

Also note that quotas are not guaranteed values. Quotas can vary based on changes from the Batch service or a user request to change a quota value.

| **Resource** | **Default limit** | **Maximum limit** |
| --- | --- | --- |
| Azure Batch accounts per region per subscription | 1-3 | 50 |
| Dedicated cores per Batch account | 90-900 | Contact support |
| Low-priority cores per Batch account | 10-100 | Contact support |
| Active jobs and job schedules per Batch account (**completed** jobs have no limit) | 100-300 | 1,000 |
| Pools per Batch account | 20-100 | 500 |

## Cores quotas in user subscription mode

If you created a Batch account with pool allocation mode set to user subscription, quotas are applied differently. In this mode, Batch VMs and other resources are created directly in your subscription when a pool is created. The Azure Batch cores quotas do not apply to an account created in this mode. Instead, the quotas in your subscription for regional compute cores and other resources are applied.

When you create a Batch account, you can specify the account configuration, which determines whether pools are allocated in a Batch service subscription (the default), or in your user subscription. If you created your Batch account with the default Batch Service configuration, then your account is limited to a maximum number of cores that can be used for processing. The Batch service scales compute nodes only up to that core limit. For this reason, the Batch service may not reach the target number of compute nodes specified by an autoscale formula.

**Pool size limits**

Pool size limits are set by the Batch service. Unlike resource quotas, these values cannot be changed. Only pools with inter-node communication and custom images have restrictions different from the standard quota.

| **Resource** | **Maximum Limit** |
| --- | --- |
| **Compute nodes in inter-node communication enabled pool** | |
| Batch service pool allocation mode | 100 |
| Batch subscription pool allocation mode | 80 |
| **Compute nodes in pool created with a managed image resource** (For pools that are not inter-node communication enabled.) | |
| Dedicated nodes | 2000 |
| Low-priority nodes | 1000 |