Understanding HPC Job Schedulers

DR. J. LAKSHMI
jlakshmi@iisc.ac.in
SERC, INDIAN INSTITUTE OF SCIENCE
BANGALORE-12

Talk Outline

- Notion of a job
- Why is it that HPC systems accept only jobs?
- Batch System
- Batch Scheduling
  - Job Queues
  - Job Priority
  - Job policy
  - Running a job
- Resource Manager
- Batch system commands

Modes of program execution

- Interactive program execution:
  - Most of us know of executing our program by typing out the executable name on the cursor prompt in a terminal window and hitting the return key!
  - The current state of the execution is observed by the contents displayed in the terminal window where program execution was initiated.
  - Most probably the output, if any, generated by the program is also displayed in this window.
  - Return to displaying the cursor prompt is an indication that the program execution is completed.
- There are some intrinsic rules associated with program execution, like env. variables, standard in-out-err devices, etc.
- Whatif there are other programs already running on the system?
  - Most HPC programs are resource intensive, efficient and maybe long running.
- Whatif your program initiates execution on multiple machines/nodes of a system?
  - Most HPC systems do not support interactive program execution!
- Interactive support on HPC setups is quite different from what normal GUI/UI based interaction experience is!

Batch execution

- Batch mode of execution refers to program execution in background.
  - Time sharing a resource across multiple jobs is possible, it may not however yield deterministic performance for all jobs.
  - Most HPC jobs can use resources of multiple nodes.
  - It makes economic sense to reduce idle time on the HPC systems, given their cost in acquiring and operating.
- Most resource policies on the HPC systems tend to be exclusive allocation based.
- User jobs are accepted and typically wait in queues before adequate resources are available to the job and then it is scheduled for execution.

What is a job?

- The user's program is not simply the name of an executable. It has input data and parameters, environment variables, descriptions of computing resources needed to run the application, and output directives. All of these specifications collectively are called a *job*.
- The user describes and submits the job to an HPC system, usually in the form of a *job script*.
- The job script contains a formal specification that requests computing resources, identifies an application to run along with its input data and environment variables, and describes how best to deliver the output data.

Job Script

- Sample Job description in job_script.sh file

```
#!/bin/bash
#PBS -l walltime=1:00
#PBS -l select=2:ncpus=2:mpiprocs=2
#PBS -N test_job
#PBS -q share
cd PBS_O_WORKDIR

mpirun ./a.out
```

chmod +x job_script.sh
qsub ./job_script.sh

Batch System

- A typical batch system is composed of three functionalities:
  - Job Scheduler or Workload Manager
    - identifies jobs to run, selects the resources for the job, and decides when to run the job
  - Resource Manager
    - identifies the compute resources and keeps track of their usage and feeds back this information to the workload manager
  - Execution Manager
    - job initiation and start of execution is co-ordinated by the execution manager of the batch system.
- Job Scheduling:
  - The batch system is responsible for receiving and parsing the job script.
  - If the job script is not correct either in expressing it's requirements (command syntax errors) or in resource specifications (custom specific), job submission fails.
  - If the job cannot be executed immediately, it is added to a queue.
  - The job waits in the queue until the job's requested resources are available.
  - The batch system then runs the job.

Batch Scheduling

- Scheduling policies:
  - **FCFS:**
    - First-come-first-serve scheduling
    - Latest job submitted is added to the bottom of the queue.
    - No other job in the queue will run before the job that is at the top of the queue.
    - Top job waits in the queue until enough jobs finish to free up the resources that it needs.
  - **Multi-priority queues**
    - Clusters with heterogeneous resources or job mixes configure multiple job queues for separation
    - Queues can be defined to support different job sizes, schedule to specific cluster resources
    - Queues can have different priorities to allow different categories of jobs to be scheduled differently
  - **Back-filling:**
    - If the scheduler has the intelligence to launch jobs lower in the queue, on resources that are currently idle, it is called a back-fill scheduler. The back-fill scheduler follows a strict rule to only schedule lower priority jobs on idle resources if it will not delay the start of the top priority job.
  - **Fair-share:**
    - A method to allocate resource shares to a user or groups of users or a project
    - A fair method for ordering jobs based on their usage history or criteria based on pricing
    - The job to be run next is selected from the set of jobs belonging to the most deserving entity
  - **Preemptive:**
    - A job with higher priority can signal currently running job to stop and release resources to allow the high priority job to run.
    - Necessary requirement for pre-emption is support for job checkpoints and restart ability.

Job Priority

- Most schedulers have another activity that further enhances their utility. The fact of the matter is that running jobs on a FCFS basis may not be fair to all users. If one user submits 100 jobs early in the morning, other users will have to wait for the jobs of that first user to complete before their jobs can run.
- Hence, modern schedulers provide a *fairness* component that re-prioritizes the queued jobs based on a system that gives an advantage to under-serviced users by moving their jobs higher in the queue.
- In addition, there are other components to prioritizing activities that consider other factors such as the size or length of the job or the job's importance or job's waiting time in the queue, etc.
- The term *size* refers to the number of nodes the job requests.
- The term *length* refers to the total time that a job wants to run.
- Job priority is derived based on job queue priority, scheduling policy, user or job queue limits, etc.

Job Queues

- While a single job queue could service an entire system, an HPC cluster is typically partitioned into pools of node resources each with its own job queue.
- Most of a cluster's node resources are dedicated to running production jobs, some nodes are typically set aside to use in debugging applications.
- These two uses, production and debugging, are at cross purposes.
  - Production runs tend to be full sized jobs that last multiple hours.
  - Debugging sessions are typically smaller sized runs and are more short lived.
  - Hence the *batch* queue is configured with wall clock and job size limits that favor production jobs.
  - The *debug* queue has shorter time and smaller size limits that ensure more immediate access to resources for smaller, quick running jobs.
- Users can specify the appropriate queue when the job is submitted or the job script specification can be parsed by the scheduler to choose the eligible batch queue for the job.

Job Policy

- The scheduler enforces *policy*.
- Each HPC computing center establishes policy or rules of what jobs can run and under what conditions.
- There are limits imposed on the size of the job, the length of time the job is allowed to run, which running jobs can be preempted and for what reasons, etc.
- The scheduler commonly provides commands for displaying the limits, access permissions, and service agreements it enforces.

Running a job

- When a job is scheduled to run, the requested compute resources are allocated to the job.
  - No other job will run on those resources during that job's execution if the allocation policy is exclusive.
  - If the allocation policy is time-shared, extent of sharing is defined during the resource configuration.
- The job script is copied to the first compute node in the allocated list and executed - and the application is launched across all the allocated nodes from the first node.
- Typically multiple tasks are launched across multiple compute cores, GPU's, and nodes - all confined to that job's allocation.

Resource Manager

- The agent of the batch system which maintains and monitors the resources managed and used by the batch system is called the *resource manager*.
- Resource manager is typically used to control resource usage policies (shared or exclusive).
- Every node under the batch system has some well defined resource attributes (like ncpus, memory, software licenses, accelerators, etc.) based on which job allocation policies are affected.
- Many batch systems allow binding of nodes to job queues based on these resource attributes.

Execution Manager

- The execution manager provides the infrastructure to control and monitor the job and collect the statistics of all the processes running all of the tasks in the job.
- These statistics are gathered, aggregated and ultimately saved to a database which will contain a record of that job's run.
- When a user's job script is crafted to launch multiple applications and/or multiple invocations of an application, each such launch is termed a *job step*.
- The statistics for each job step are also captured and recorded.

Batch system commands

- The batch system provides a collection of commands for users to interact with the scheduler and resource manager.
- There are commands to submit a job, display the job queue, and see status and details of the job itself.
- In addition, there are commands which provide information on the computing resources, showing which resources are allocated, which are idle, and which are off-line or down.

Popular Batch scheduling software

- Commercial software
  - PBSPro - M/s Altair
  - LSF - M/s IBM
  - LoadLeveller - M/s IBM
- Opensource
  - Condor: https://research.cs.wisc.edu/htcondor/
  - Torque: https://www.adaptivecomputing.com/products/torque/
  - Slurm: https://slurm.schedmd.com/overview.html

PBSPro Commands

- qmgr
  - Configuration
- qsub
  - Submit Jobs
- qstat
  - View Status of Jobs/Queues/Servers
- qrls/qhold
  - Hold/Release Jobs
- qdel
  - Delete submitted jobs

email:
jlakshmi@iisc.ac.in

Any questions?
Thankyou!