What is “overutilization of resources” on HPCC?

By Xiaoge Wang, iCER Research Consultant

The consequences of “over utilizing HPCC resources” are high. Anyone who has done this before will tell you that it’s no fun having your job terminated in the middle of execution or having your account blocked from submitting jobs to the HPCC cluster. Overutilization of resources is not a rare occurrence nor is it limited to users from a specific field/discipline. It can happen to any of us if we are not careful in submitting jobs. This past April, iCER had a dozen tickets related to resource overutilization. In this article, we’ll share some tips regarding the resource usage and how to avoid overutilization.

Resource type	Consequence of over utilizing	Side effects	Prevention	Correction
Memory	Batch jobs: terminated automatically by resource manager. Interactive jobs: experience slowdown.	Batch job: job killed so no impact on other users. Interactive jobs: slow down other jobs on the same node	Monitor the job’s memory usage and get better estimation of memory required.	Modify the line in batch job script #PBS -l mem=...
Walltime	Batch jobs: terminated automatically by resource manager. Long job on dev node may be terminated.	Batch job: job killed so no impact on other users.	Monitor the progress of the program and get better estimation of walltime.	In your job script modify the line: #PBS -l walltime=.....
CPU	Job may be terminated by system administrators. Job owner may be blocked from submitting jobs until jobs are tested safe.	Slow down other jobs running on the same node or crash the node	Monitor the CPU usage of your jobs.	In your job script modify the line: #PBS -l nodes=n:ppn=m.

Resource type	Consequence of over utilizing	Side effects	Prevention	Correction
Disk space	Job may be terminated by system administrators. Job owner may be blocked from submitting jobs until jobs are tested safe.	May slow down or crash the file system server. All users running jobs on the same file system space are affected.	Avoid large quantity of standard output. Use local disk if possible. Reduce number of I/O operation by increasing the I/O data size of each operation.	Modify your program to reduce the number of I/O operations. Reduce the standard output. Redirect the standard output to a file.

Table 1: Summary of the type of resources, consequences of over utilizing each type, as well as how to prevent and correct overutilization for batch jobs.

In general, “overutilization of resources” may occur on development nodes when a user launches many processes or launches a process that uses many threads for an extensive period. On a compute node, this occurs when a batch job utilizes more resources than requested at the submission time. Table 1 summarizes the type of resources, the consequences of overutilization of each type, as well as how to prevent and correct overutilization for batch jobs.

For reference, the following frequently asked questions (FAQs) and answers related to resources overutilization are provided below.

How do I run my job on a development node without over utilizing it?

Answer: Although there are no specific restrictions on running your program on development (dev) nodes, we do have the following common practices:

1) Run your application on a dev node with low loads and closely monitor the usage of the node you are using. When you connect to the gateway node, check the login message and ssh to nodes where the load is low. When you are on a dev node, use the “top” command to see the summary of the node and the list of processes running on that node. If the “load average” returned is greater than the number of physical cores on the node (20 for intel14, 28 for intel16), the node is fully loaded. If this number is greater than doubled, the node is over loaded and processed that put the largest load on the node may be terminated by system administrators. If the load on a node becomes high, try to use another node. If all development nodes are overloaded, please report the situation to iCER.
2) Do not run long processes on develop node. Try to limit to 2 hours.
3) Do not use a large percentage of available cores, especially when load is high. Use “top” or “ps” command to monitor your usage of the resources.

How do I monitor resource usage of my batch job?

Answer: Assume that you have a job running with the ID: `<JOB_ID>`. Retrieve the job ID by using: `qstat -u <NetID>` or `showq -u <NetID>` where <NetID> is your username on HPCC. To monitor the usage of the job use the command: `checkjob -v <JOB_ID>`. The output of this command, provides the following information about the job:

1) "Total Requested Tasks" and "TaskCount": This is the number of tasks the job requested.
2) "Dedicated Resources Per Task": This is what the scheduler allocated for the job. It includes the number of processors and memory.
3) "Utilized Resources Per Task": shows the number of processors utilized and the memory utilized per task. This number should be smaller than "Dedicated Resource Per Task".
4) "Average Utilized Procs": Shows the average number of processors utilized. This number should not be much larger than "TaskCount" times "Dedicated Resource Per Task".

If "Utilized Resources" is greater than "Dedicated Resources", the job is over utilizing resources. The greater the number of over utilized procs (compared to dedicated procs), the more resources the job is over utilizing. To stabilize an over utilized node, jobs with higher overutilization should be deleted first, either by the user or a system administrator.

Some common mistakes that lead to overutilization of resources:

1) **Memory Requested:** For example, in the following job script:
   #PBS -l nodes=2:ppn=3,mem=2GB,walltime=2:00:00
   the number of tasks in the job is 2*3=6 and "mem=2GB" specifies the total memory of the job. Thus, the requested memory per task is 2GB/6, not 2GB.
2) **Nodes and ppn requested:** Only programs that have capability for inter-node communication, such as MPI, should use multiple nodes or interchange the number of nodes with the number of ppn. Parallel programs that only support multithread/multicore should only use a single node with multiple ppn, and the number of ppn should not be greater than the number of physical cores on the node. If your program does not have the capability of cross nodes parallel, requesting nodes=n, where n>1 may lead to a situation in which the resource manager allocates the job to multiple nodes when the job only runs computation on a single node; thus, overloading this node and wasting the other nodes requested. Users need to know the program before preparing the job script. To monitor your multi-node job, use the `qstat -f <Job_ID>` command and look for the line that begins with "`exec_host = …`" to obtain the names of nodes (`<hostname>`) assigned to the job. To see all the processes belonging to you on a specific node (`<hostname>`), type: `ssh <hostname> ps -u <NetID>` where (your NetID is used to identify the owner of the process).
3) **Interactive Jobs:** When you request an interactive job and the scheduler successfully allocates you a node, you are brought to the terminal interface of the allocated compute node. Although the interface looks similar to a dev node, this node is managed by the resource manager. Thus, if you over utilize the memory or walltime, your session is terminated. Monitoring the usage of cores and disk space for an interactive job, is similar to doing so on a dev node. Users are responsible for monitoring the usage and not over utilizing the node. If you requested a multi-node interactive job, ensure that you understand how to launch/spawn processes to other dedicated nodes.

If you have any questions regarding how to avoid over utilizing dev nodes or compute nodes, contact iCER at https://icer.msu.edu/contact.