Grid Job Memory Requirements

When submitting the qsub or qlogin request, you should use flags to request the amount of RAM that you expect to need, so you specifically get handed to a machine with that amount of RAM free, and do not end up getting handed to a machine that's already overextended, and run out of memory. You can also use hard and soft limits to gently or forcibly kill your own program in the event that it exceeds the expected amount of memory.

If you don't specify limits, a default s_vmem value of 2G will be set.

Please use the following qsub options to request memory need and to set soft and/or hard limits:

```
-l mem_free=MEM_NEEDED
-l h_vmem=MEM_MAX
-l s_vmem=MEM_MAX
```

where MEM_NEEDED is the amount of memory (in megabytes M, or gigabytes G) that your job will require and MEM_MAX is the upper bound on the amount of memory your job is allowed to use. mem_free and swap_free are requestable complexes — so you can request, in your job script, that your jobs are only handed to nodes with more than X amount of memory free.

Keep in mind that the soft limit s_vmem will pass a SIGINT or SIGUSR1 to the program when it hits that memory usage, and the hard limit h_vmem will outright kill the program when it hits that memory usage. NOTE: The hard limit may prematurely kill programs that like to probe all available memory when they first start up, such as MATLAB and all MPI programs! You can solve this with MATLAB by setting a ulimit, but with MPI, you should just use soft limits and make sure that your program respects them.

For example, if your job will require 4GB of memory and you want to receive a signal at 4G usage, and if it doesn't respond to the signal, you want the program to be killed forcibly at 6G usage, you could type:

```
qsub -cwd -l mem_free=4G,s_vmem=4G,h_vmem=6G batch.sh
```

(Remember that you can put the -l options in your .sh script rather than specifying them on the command line.)

To see within a shell (interactive or batch) what limits are in place, run `ulimit`, for example:

```
ulimit -a
core file size          (blocks, -c) unlimited
data seg size           (kbytes, -d) unlimited
scheduling priority             (-e) 0
```

```
file size               (blocks, -f) unlimited
pending signals                 (-i) 257184
max locked memory       (kbytes, -l) unlimited
max memory size         (kbytes, -m) unlimited
open files                      (-n) 4096
pipe size            (512 bytes, -p) 8
POSIX message queues     (bytes, -q) 819200
real-time priority              (-r) 0
stack size              (kbytes, -s) unlimited
cpu time               (seconds, -t) unlimited
max user processes              (-u) 4096
virtual memory          (kbytes, -v) 2097152
file locks                      (-x) unlimited
```

Parallel Jobs

For parallel jobs (like for MPI) the grid will allocate resources on a per-slot basis.

For example, if you use `-l s_vmem=4G` and `-pe openmpi 4`, it will set a separate 4GB soft limit on each of the 4 processes, whether they are on the same physical machine or not. The total memory allocation and limit for the job will be 16 GB.

For the `threaded` parallel environment, all processes *will* be on the same machine, so there will effectively be a memory limit on that machine of the s_vmem size multiplied by the number of slots requested. If no single machine has that amount of memory available at the time, the job will wait, but if no machine has that much in total, the job will be rejected with the message `error: no suitable queues` .

Checking available memory

You can use `qstat -F` to check for available resources. If you include `-F` but leave out the list of resources it will list values for all resources. To see the three memory resources described here, plus the total amount of RAM detected, for bme.q:

```
$ qstat -F mem_free,s_vmem,h_vmem,mem_total -q bme.q
queuename                      qtype resv/used/tot. load_avg arch
states
-------------------------------------------------------------------------------
-
bme.q@bme-compsim-1.bu.edu     BIP   0/20/20        0.01     linux-x64
        hl:mem_total=62.809G
        hl:mem_free=61.985G
        hc:s_vmem=24.000G
        qf:h_vmem=infinity

...
```

For bme-compsim-1, the total amount of RAM detected is about 63 GB. The amount of memory actually available according to the operating system is about 62 GB. The grid is reserving some memory for each of the 20 used job slots, so the available s_vmem is down to 24 GB. h_vmem is only used for setting an upper limit on usage for a job and isn't tracked, so it always shows infinity.

The two-letter code at the start of each resource entry shows how the grid came up with that value. For the first letter:

- g – global setting
- q – queue setting
- h – host setting

And the second letter:

- l – detected load value (via the OS)
- c – consumable value (tracked by the grid)
- f – fixed limit

For more information on resources see `man qstat`, particularly the "Full Format" section.

To see the resources defined by the grid for a particular host, look at the complex_values section of the `qconf -se HOST` output, like:

```
qconf -se bme-compsim-1 | grep complex_values
complex_values        s_vmem=64G
```

So, the grid knows bme-compsim-1 has 64 GB RAM total, and will deduct its allocations from that.