

Running MPI across multiple machines using Sun Grid Engine (SGE)

RDC 2/11/2009

These instructions describe how to setup MPI runs across multiple machines using the Sun Grid Engine (SGE) job queue infrastructure. Jobs submitted to the queue will be executed on lab machines that do not have someone currently logged into the console.

Setup

First, several items need to be dealt with prior to submitting a job to the queue. This need be done only once and should not need to be repeated.

Touch a file in your home directory called ".smpd". Make this file `chmod 0600`.

Create a file `.rhosts` in your home directory, `chmod'd 0600`, with these contents:

```
n401.int.cec.wustl.edu username
n402.int.cec.wustl.edu username
n403.int.cec.wustl.edu username
n404.int.cec.wustl.edu username
n405.int.cec.wustl.edu username
n406.int.cec.wustl.edu username
n407.int.cec.wustl.edu username
n408.int.cec.wustl.edu username
n409.int.cec.wustl.edu username
n410.int.cec.wustl.edu username
n411.int.cec.wustl.edu username
n412.int.cec.wustl.edu username
n413.int.cec.wustl.edu username
n414.int.cec.wustl.edu username
n415.int.cec.wustl.edu username
n416.int.cec.wustl.edu username
n417.int.cec.wustl.edu username
n418.int.cec.wustl.edu username
n419.int.cec.wustl.edu username
n420.int.cec.wustl.edu username
n421.int.cec.wustl.edu username
n422.int.cec.wustl.edu username
n423.int.cec.wustl.edu username
n424.int.cec.wustl.edu username
n425.int.cec.wustl.edu username
n426.int.cec.wustl.edu username
n427.int.cec.wustl.edu username
n428.int.cec.wustl.edu username
n429.int.cec.wustl.edu username
n430.int.cec.wustl.edu username
rote.cec.wustl.edu username
gridengine.int.cec.wustl.edu username
```

Replace `username` in the above contents with your login name.

The setup is now complete. The description below covers how to prepare a job for execution using the SGE queuing system.

Job Preparation

MPI jobs that are to be executed under SGE must be linked with different libraries than those to be executed on the local machine. To accomplish this, use the command `mpicc.sge` (rather than the alternative `mpicc.local`) as the compiler. An example of this usage is provided in the `makefile` for problem 1 of assignment 2 (available on the class web page). In the example, two separate (distinctly named) binaries are created. One is for local execution, the other is for execution on the queue. Both are compiled from the same source code, the difference is which variant of the compilation is invoked.

To invoke MPI, we use `mpiexec.sge`; however, it is most effective to invoke it using a shell script. An example script is included on the class webpage, called `monte_pi_mpi.sh`, and it includes internal instructions how to alter it for various purposes (e.g., changing the number of processes, changing the compiled executable to startup, etc.).

Submitting Jobs

Once a suitable shell script has been prepared, it is submitted to the queue. During execution you can monitor the queue. Once execution is complete, 4 files will be placed in your current working directory: `stdout` and `stderr` from your program plus `stdout` and `stderr` from the queuing system.

To submit to the queue, first `ssh` into the machine `gridengine.int.cec.wustl.edu` and issue the command `qsub <script>`, where `<script>` is the shell script prepared above (e.g., `monte_pi_mpi.sh`).

To query the status of your job (and all jobs in the queue), issue the command `qstat`.

Jobs can be deleted from the job queue using `qdel`. Note that there are `man` pages available for `qsub`, `qstat`, and `qdel`.

The four files created in your current working directory are named `<jobname>.o<jobnumber>` (which is `stdout` from your program), `<jobname>.e<jobnumber>` (which is `stderr` from your program), `<jobname>.po<jobnumber>`, and `<jobname>.pe<jobnumber>` (which are `stdout` and `stderr` from the queuing system). The two `stderr` files should normally be empty. You can examine the queuing system output file to determine the machines used.

IMPORTANT!!! You will have limited ability to debug codes running under SGE. If it hangs, it will likely hang the entire queue. It will take an administrator to clear the queue. Your colleagues will not be happy with you! Test your code locally (even scaling up the number of processes to larger than the number of processors on the local machine) before running it under SGE. Run under SGE only for timing information once you are confident it is correct.