1 Running Qs on the FBS GNU/Linux cluster

This document is intended to be a short how-to guide to aid running Qs on the FBS cluster. It can not be used as a substitute for the main documentation. Running Qs on the cluster is pretty-much straightforward with one very significant exception : the current parallel version of Qs does not scale well to more than 5-15 processors. The implication is that if you lightheartedly start a parallel job utilising, say, 120 processors, Qs will most probably run slower than it would run on a stand-alone machine.

In summary the plan for running Qs on the cluster is :
* Determine 'best' number of processors to be used for the given problem.
* Submit as many independent jobs as the number of minimisations you want to perform.

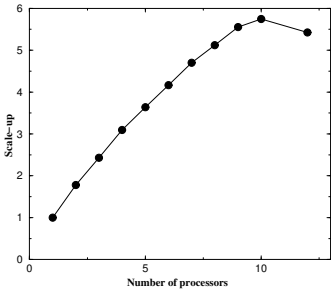
If, for example, you found that the best number of processes for your problem is 9 (corresponding to a scale-up of, say, 6.0) and you want to perform 5 minimisations, then you could submit 5 jobs (each occupying 9 processors) and you would expect your calculations to finish in 1/30th of the time they would have taken on a single processor.

2 'Best' number of processors : what to expect

The best number of processors for any given problem depends mainly on two parameters : the number of unique reflections and the number of crystallographic symmetry operators (for the given space group). In general, the program will perform better as the number of unique reflections decreases and the symmetry increases. To get an idea of what to expect for your problem, I wasted some of FBS's CPU time to run a few tests with problems of different sizes. In all cases, I used short runs (200,000 steps each) and I assumed that the machines were otherwise idle. The results are shown below (wallclock times in seconds, some graphs are shown twice to aid comparison) :

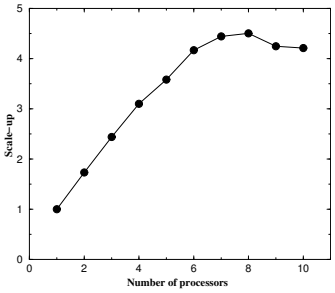
9383 reflections, P422 :

Processors	Wallclock	Scale-up
1	8421	1.0
2	4733	1.7792
3	3464	2.4310
4	2723	3.0925
5	2315	3.6375
6	2021	4.1667
7	1792	4.6992
8	1644	5.1222
9	1516	5.5547
10	1466	5.7442
12	1553	5.4224



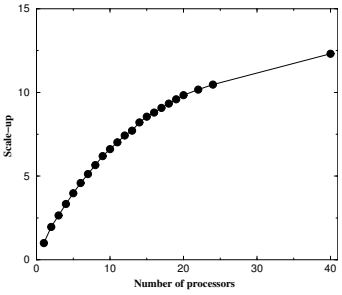
18698 reflections, P422 :

Processors	Wallclock	Scale-up
1	15396	1.0
2	8888	1.7322
3	6311	2.4395
4	4967	3.0996
5	4297	3.5829
6	3696	4.1655
7	3466	4.4420
8	3418	4.5043
9	3625	4.2471
10	3656	4.2111



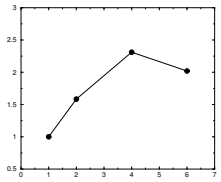
4692 reflections, P422 :

Processors	Wallclock	Scale-up
1	4909	1.0
2	2427	1.9636
3	1850	2.6535
4	1474	3.3303
5	1236	3.9716
6	1069	4.5921
7	959	5.1188
8	867	5.6620
9	792	6.1982
10	743	6.6069
11	700	7.0128
12	661	7.4266
13	636	7.7185
14	598	8.2090
15	574	8.5522
16	558	8.7974
17	541	9.0739
18	526	9.3326
19	512	9.5878
20	499	9.8376
22	483	10.1635
24	469	10.4669
40	399	12.3032



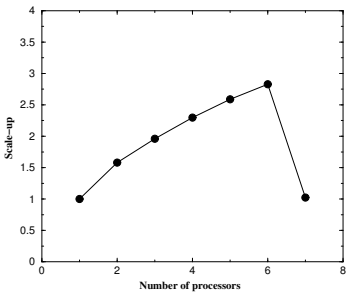
18698 reflections, P21 :

Processors	Wallclock	Scale-up
1	4405	1.0
2	2780	1.5845
4	1906	2.3111
6	2181	2.0197



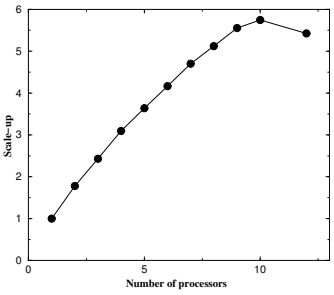
9383 reflections, P21 :

Processors	Wallclock	Scale-up
1	2386	1.0
2	1510	1.5801
3	1218	1.9589
4	1039	2.2964
5	922	2.5878
6	844	2.8270
7	2330	1.0240



9383 reflections, P422 :

Processors	Wallclock	Scale-up
1	8421	1.0
2	4733	1.7792
3	3464	2.4310
4	2723	3.0925
5	2315	3.6375
6	2021	4.1667
7	1792	4.6992
8	1644	5.1222
9	1516	5.5547
10	1466	5.7442
12	1553	5.4224



3 Determining the 'best' number of processors

To determine the best number of processors for your problem :

* Run the program on a stand-alone machine using the automatic mode and stop it after in starts the actual minimisation (as described in the program's documentation). The program will create a file named 'Qs_auto.in' in its current directory.

* Copy the files Qs_auto.in, data.hkl and model.pdb (or model1.pdb, model2.pdb, etc) to your area on the cluster. Edit the file Qs_auto.in and change the lines saying

```
CYCLES          5
STEPS           10000000
```

to

```
CYCLES          1
STEPS           200000
```

(the actual number of steps you will find in Qs_auto.in depends on the number of molecules per asymmetric unit and may be different from the one shown above).

* Prepare a file with the name, say, 'Qs_lam.sh' containing the following :

```
#!/bin/bash
#
# The first of the lines that follow (the one saying "#$ -pe rack1 12")
# defines two things :
# 1. The number of processors to use (12 for this example)
# 2. The queue ('parallel environment') that the job will run on. On
#    'rack1' you can use up to 76 CPUs concurrently, with 'rack2' 72.
#    This separation in two queues is necessary due to network topology.

#$ -pe rack1 12
#$ -j y -o Qs.LOG
#$ -cwd

#
# The line that follows defines the current working directory
#
cd /home/bmbnmg/last

#
# The actual MPI call to run the parallel version of Qs
#
/usr/local/lam-pgi_5.1/bin/mpirun -np $NSLOTS /fbs/software/Qs/Qs_MPI Qs_auto.in
```

* Create directories with names like 1proc/ 2proc/ 4proc/ etc and copy in each one of them the Qs_auto.in, data.hkl, model.pdb and Qs_lam.sh files.

* Successively enter each of these directories, edit the Qs_lam.sh file, change the number of processors to be used (e.g. make it 4 for the file located in the 4proc/ directory), change the directory name, and submit a job with 'qsub Qs_lam.sh'.

* After the jobs finish, find the wallclock time recorded for each of the runs. This is written out by the program after each minimisation finishes (do a tail -50 Qs.LOG and you should see it). Error messages of the type "FFTW failed to read the wisdom file ..." can safely be ignored.

* Decide how many processors per minimisation you will use.

4 Production runs

* Prepare directories like minim1/ minim2/ ... minim5/ and copy the Qs_auto.in, data.hkl, model.pdb and Qs_lam.sh files in them.

* Edit the Qs_auto.in file, change the number of steps to its original value (say, 10,000,000), keep the number of cycles to the value of 1, find the line saying 'SEED 147579' and change it to a different (integer) number. The values for SEED should be different for each of your minimisations, otherwise you will be performing a large number of identical minimisations.

* Edit the Qs_lam.sh file, define the number of processors to be equal to the 'best' you have already determined and change the working directory path name.

* Submit the jobs.

Good luck,
NMG, September 2004