Cheat Sheet PBS Scheduler, SSH connections and Valgrind

PBS User Commands

Command	Usage	Description	Options	Option Description	General Observations
qsub	qsub <options> [script]	Submit a pbs job	-I	Submit an interactive job	
			-IX	Submit an interactive job with X forwarding	To exectue this option a remote login with display X11 is necessary (ssh -X)
qstat	qstat <options>	Print job status	-a	Show all jobs	*showq* is an alternative command that shows the jobs status of all users in an alternative output
			-n	Show all jobs with node information	
			-f <jobid>	Show full information for a given job	
			-u <userid>	Show all jobs for userid	
			-r	Show running jobs	
			-q	Show information about queues	
			-Qf <queue>	Show full information about queue	
			-x	Show jobs hystory	
			-B	Show summary status of the job server	
qdel	qdel <jobid>	Delete pbs batch job	-W force	Force kill a job	
qhold	qhold <jobid>	Hold a job			
qrls	qrls <jobid>	Release a job			
qmove	qmove <new_queue> <jobid>	Moves PBS batch job between queues			
showstart	showstart <jobid>	Shows the approximate starting time of the job			
checknode	checknode <nodeid>	Shows the status of the node			

PBS Jobs Attributes

PBS job attributes can be set in two ways:
1. as command-line arguments to qsub, or
2. as PBS directives in a control script submitted to qsub. example: #PBS -q batch ('#PBS' it is mandatory)

Attribute	Usage	Description	Options	Option Description	General Observations
-q	-q <queue>	Specifying queue and/or server			
-e	-e <path/error_file_name>	Specifying path for error files			
-o	-o <path/output_file_name>	Specifying path for output files			
-j	-j oe	Merging output and error files			
-N	-N <job_name>	Specifying a job name			
-M	-M <mail@mailserver>	Setting email recipient list			
-m	-m <options>	Specifying email notifications	a	Notify when reservation is terminated	You must not put white spaces between options
			b	Notify when reservation period begins	
			c	Notify when reservation is confirmed	
			e	Notify when reserva-tion period ends	
			n	Do not send mail. Can not be used with other letters	
-l	-l <resource_option_request>	Requesting job resources	nodes=#:ppn=#	Declares the node configuration for the job.	Specific processor arquitecture can be requested by adding the corresponding flag (intel, mpi, amd)
			walltime=hh:mm:ss	Specifies the estimated maximum wallclock time for the job	
			cput=hh:mm:ss	Specifies the estimated maximum CPU time for the job	
			mem=#	Specifies the estimated maximum amount of RAM used by job followed by b, kb, mb, or gb. By default, the integer is interpreted in units of bytes	
			vmem=#	Specifies the estimated maximum amount of virtual memory used by job followed by b, kb, mb, or gb. By default, the integer is interpreted in units of bytes	
			ncpus	Declares the number of CPUs requested	
-p	-p <job_priority>	Integer between -1024 and +1023. Defines the priority of the job. Higher values correspond to higher priorities.			

PBS Environment Variables

The following is a list of the environment variables set by PBS for every job. (common use, $VARIABLE)

Variable	Description	General Observations			
PBS_ENVIRONMENT	set to PBS_BATCH to indicate that the job is a batch job; otherwise, set to PBS_INTERACTIVE to indicate that the job is a PBS interactive job				
PBS_JOBID	Job identifier given by PBS when the job is submitted.	Created upon execution.			
PBS_JOBNAME	Job name given by user.				
PBS_NODEFILE	The filename containing a list of vnodes assigned to the job.				
PBS_O_WORKDIR	Absolute path to directory where qsub is run.	Value taken from user's submission environment.			
TMPDIR	Pathname of job's scratch directory				
PBS_JOBDIR	Pathname of job's staging and execution directory on the primary execution host.				
PBS_QUEUE	Name of the queue from which the job is executed				
PBS_O_LOGNAME	Name of submitting user				
PBS_NP	Number of execution slots (cores) for the job				
PBS_O_HOST	Name of the host upon which the qsub command is running				

SSH Usser commands

SSH (Secure SHell) is a network protocol which provides a replacement for insecure remote login and command execution facilities, such as telnet, rlogin and rsh. SSH encrypts traffic in bothdirections, preventing traffic sniffing and password theft.

Command	Usage	Description	Options	Option Description	General Observations
ssh	ssh <hostname>	Connect to remote host	ssh -l <user> <hostname>	Connect to remote host as a specific user	The first time your client connects to a ssh server, it asks you to verify the server's key. (*The authenticity of host 'hachi.mindrot.org (203.36.198.102)' can't be established.RSA key fingerprint is cd:41:70:30:48:07:16:81:e5:30:34:66 :f1:56:ef:db.Are you sure you want to continue connecting (yes/no)?* ***yes***)
			ssh <user@hostname>		
			-p <portid>	Connect to remote host on a non-standard port	
			-t <command>	Execute commands remotely (example: ssh -t user@hostname vi /tmp/test)	
			-i <identity_file>	Connect using an identity file (SSH keypair) for authentication	
			-C	Compress traffic between hosts. (default: no)	
			-X	Display X11 graphical programs from your remote host on the local host. (default: no)	
scp	scp <user@host:/path/to/source/file> </path/to/destination/file>	This is the original SSH file transfer mechanism	-r	Copy files recursively (example: copy a directory)	
rsync	rsync <options> <user@host:/path/to/source/files> </path/to/destination/files>	Put two sets of files into synchronisation	-e	Just list files on the remote machine, in a particular directory	
			-ve	To synchronise/copy a remote set of files to a local set	

Valgrind User Commands

Valgrind is designed to be as non-intrusive as possible. It works directly with existing executables. You don't need torecompile, relink, or otherwise modify the program to be checked.

Command	Usage	Description	Options	Option Description	General Observations
valgrind	valgrind <valgrind-options> your-prog <your-prog-options>	Invoke Valgrind	--tool	Dictates which Valgrind tool to run	
			--gen-suppressions	Print out a suppression for each reported error	
			-fno-inline	Reduce confusion when navigating around large C++ apps (recommended for C++ large codes)	
			--read-inline-info=<yes|no>	Read the debug information describing inlining information	
			--leak-check=full	Determines if the block is reachable from pointers within the root-set.The root-set consists of (a) general purpose registers of all threads, and (b) initialised, aligned, pointer-sized data words in accessible client memory, including stacks.	
			--help		
			-q	Run silently, and only print error messages.	
			--vgdb=<no|yes|full>	Allows an external GNU GDB debugger to control and debug your program when it runs on Valgrind	
			-v	Be more verbose. Gives extra information on various aspects of your program, such as: the shared objects loaded, the suppressions used, the progress of the instrumentation and execution engines, and warnings about unusual behaviour.	

Valgrind Tools

Valgrind tools must be set as command-line arguments to --tool=<tool> Valgrind option

Tool	Description	General Observations			
Memcheck	Is a memory error detector. It helps you make your programs, particularly those written in C and C++, more correct.				
Cachegrind	Is a cache and branch-prediction profiler. It helps you make your programs run faster.				
Callgrind	Is a call-graph generating cache profiler. It has some overlap with Cachegrind, but also gathers someinformation that Cachegrind does not.				
Helgrind	Is a thread error detector. It helps you make your multi-threaded programs more correct.				
DRD	Is also a thread error detector. It is similar to Helgrind but uses different analysis techniques and so may finddifferent problems.				
SGcheck	Is an experimental tool that can detect overruns of stack and global arrays. Its functionality is complementary to that of Memcheck: SGcheck finds problems that Memcheck can't, and vice versa.				