Brigham Young University

Fulton Supercomputing Lab

Ryan Cox

SLURM User Group 2013

Fun Facts

- ~33,000 students
- ~70% of students speak a foreign language
- Several cities around BYU have gigE at home
- #6 Top Entrepreneurial Programs: Undergrad (Princeton Review)
- Many BYU grads go on to write schedulers
- #1 Stone Cold Sober – 15 years running (Princeton Review)
- #1 on "*25 Colleges Where Students Are Both Hot And Smart*" (Business Insider / College Prowler)

Staff

- 4 FTEs
  - Operations Director
  - 2 x Systems Administrator
  - Applications Specialist / User Support
- 4 Students
  - Hardware Technician
  - Web Developer
  - 2 x Applications Specialist

Organization

- Supercomputing reports to CIO
- Support BYU, BYU-Idaho, BYU-Hawaii
- Free access for faculty, grads, undergrads, collaborators
- Large number of undergrad research assistants

Compute Hardware

- m6 - 320 Dell M620 blades
  - Dual eight core Sandy Bridge (2.2 GHz)
  - Infiniband
- m7 - 512 Dell M610 blades
  - Dual six core Westmere (2.67 GHz)
  - Gigabit Ethernet
- 8 Dell M910 blades (256 GB RAM each)
- 4 Privately-owned Dell blade enclosures (52 x M610's)
- A few GPUs, Xeon Phi, other assorted hardware
- Total: 12,100 cores

Using SLURM since January

- Switched to SLURM from Moab/Torque in January
- Commercial support from SchedMD
- Very tight timeline to switch due to license expiration and a hectic schedule
- No desire to immediately retrain users on SLURM

Transition to SLURM

- Split-brain, rolling upgrade to SLURM from Moab/Torque
  - Moved nodes to SLURM as jobs freed and queue drained
- Wrapper scripts: $jobid < 4000000? That's a SLURM job!
  - SLURM? Use SLURM wrapper
  - Moab? Call real Torque/Moab command
- Heavily modified SLURM's qsub wrapper to work with our installation, should have written from scratch. ~99% compat.
- Wrote Moab wrappers (not contrib-worthy code, trust me)*

* Contact me if you're not scared off by hacked-together PHP code from our web developer that we use in production... it does work but we don't want our names attached to it :)

What they don't know won't hurt them

- Users worry about change, why give advance notice?
- No notification whatsoever to users before switch to SLURM*
- Email from us: “New jobs go to SLURM, your scripts and the PBS commands stay the same. Running jobs keep running”
- Transition went well
- Most users oblivious, others excited to try SLURM tools
- Excellent support from SchedMD
  - Few bugs
  - Bugs typically patched within hours

* *Yes, we are that crazy*

General policies (1 of 2)

- Max walltime is 16 days. Will reduce to 7 days in January
- What is the max walltime at your site?
- Shared node access
  - Users must request memory. Enforced w/cgroups
  - pam_namespace creates temporary /tmp and /dev/shm per user*
  - Future: require disk allocation & use quotas?
- Defaults: 30 min timelimit, 512M mem/core, 1 core
- Each PI has a SLURM account, all accounts equal

* http://tech.ryancox.net/2013/07/per-user-tmp-and-devshm-directories.html

General policies (2 of 2)

- GrpCPURunMins per account
  - Staggers the job start/end times
  - Encourages shorter jobs
- No maximum node/job/core count per user or account
- Ticket-Based multifactor (previously multifactor2)
- Feature-based scheduling: no requesting queue/partition

Feature-based scheduling

- Users select necessary node features
  - ib, avx, sse4.2, sse4.1
- Features + Lua script limits which partitions are available to the job
- Least capable nodes are prioritized
- Users don't have to watch utilization of each partition; better load balancing

Job Submit Plugin

- all_partitions plugin lists all partitions for lua to examine (subject to AllowGroups)
- If special “empty” partition is present, lua script knows the user didn't request a specific partition
- Remove any partitions they can't or shouldn't run in
- Example: Allow access to big memory nodes if the job needs that much memory, deny partition access if not

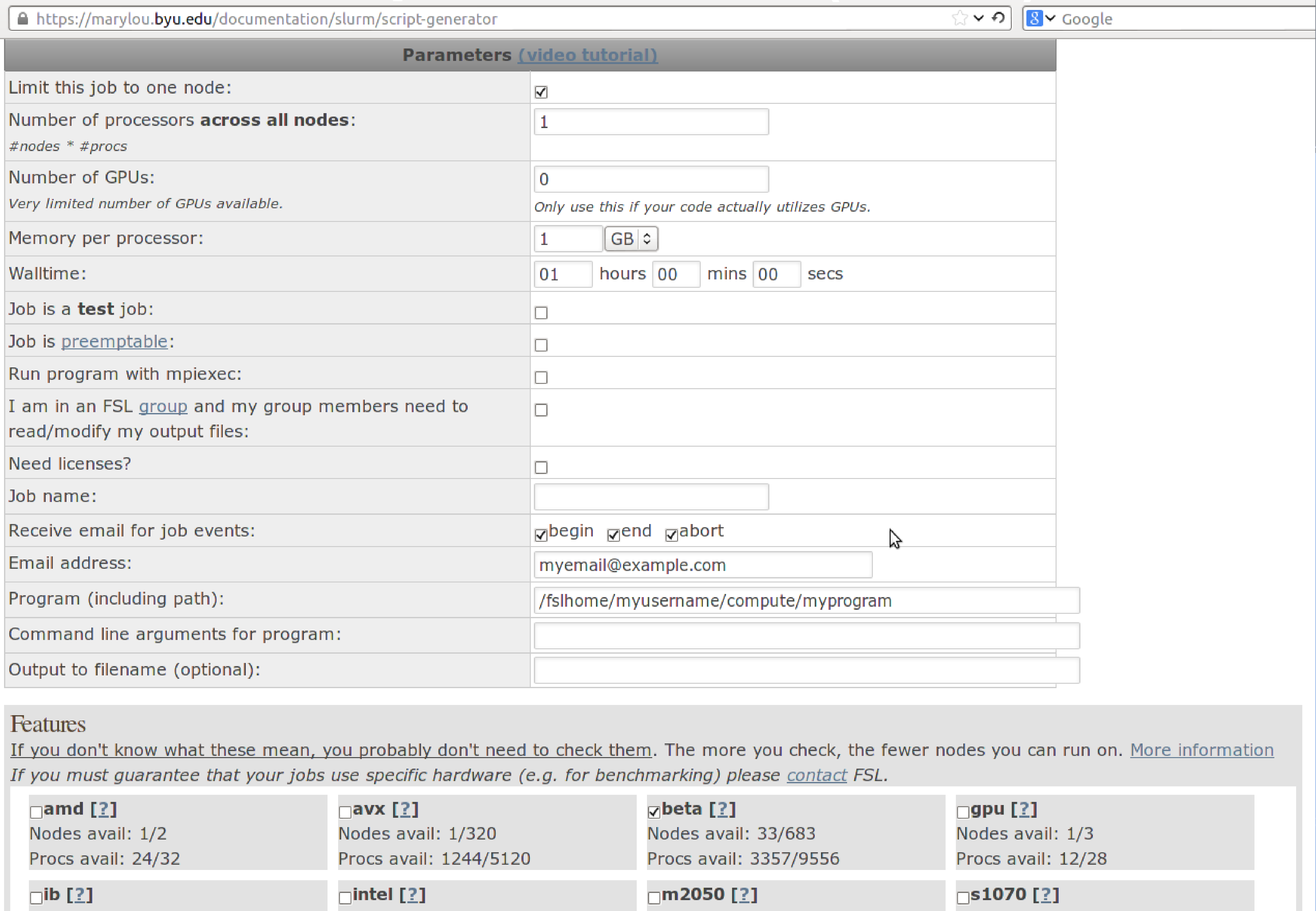
Transient node failures

- We miss Torque's ERROR handling on compute nodes
- Filesystem check timed out? That *should* clear soon
- Drain/resume tracking of transient failures + real hardware problems + others: too complex
- Health check scripts create 10 minute reservations
- Scripts run at least once every ten minutes

User Experience

- Wrote “whypending” tool to make obvious SLURM messages even more obvious. Shows partial/full idle count within partition, taking into account memory req
- Web services API
- WIP: Custom script parses Gaussian params and others to submit sane resource requests
- 2-5 minute training videos on YouTube channel
- Web-based Script Generator (SLURM/PBS)
  - https://marylou.byu.edu/documentation/slurm/script-generator

Script Generator (1 of 2)



https://marylou.byu.edu/documentation/slurm/script-generator

Parameters (video tutorial)

Limit this job to one node:	☑
Number of processors **across all nodes**: *#nodes * #procs*	1
Number of GPUs: *Very limited number of GPUs available.*	0 *Only use this if your code actually utilizes GPUs.*
Memory per processor:	1 GB
Walltime:	01 hours 00 mins 00 secs
Job is a **test** job:	□
Job is preemptable:	□
Run program with mpiexec:	□
I am in an FSL group and my group members need to read/modify my output files:	□
Need licenses?	□
Job name:	
Receive email for job events:	☑begin ☑end ☑abort
Email address:	myemail@example.com
Program (including path):	/fslhome/myusername/compute/myprogram
Command line arguments for program:	
Output to filename (optional):	

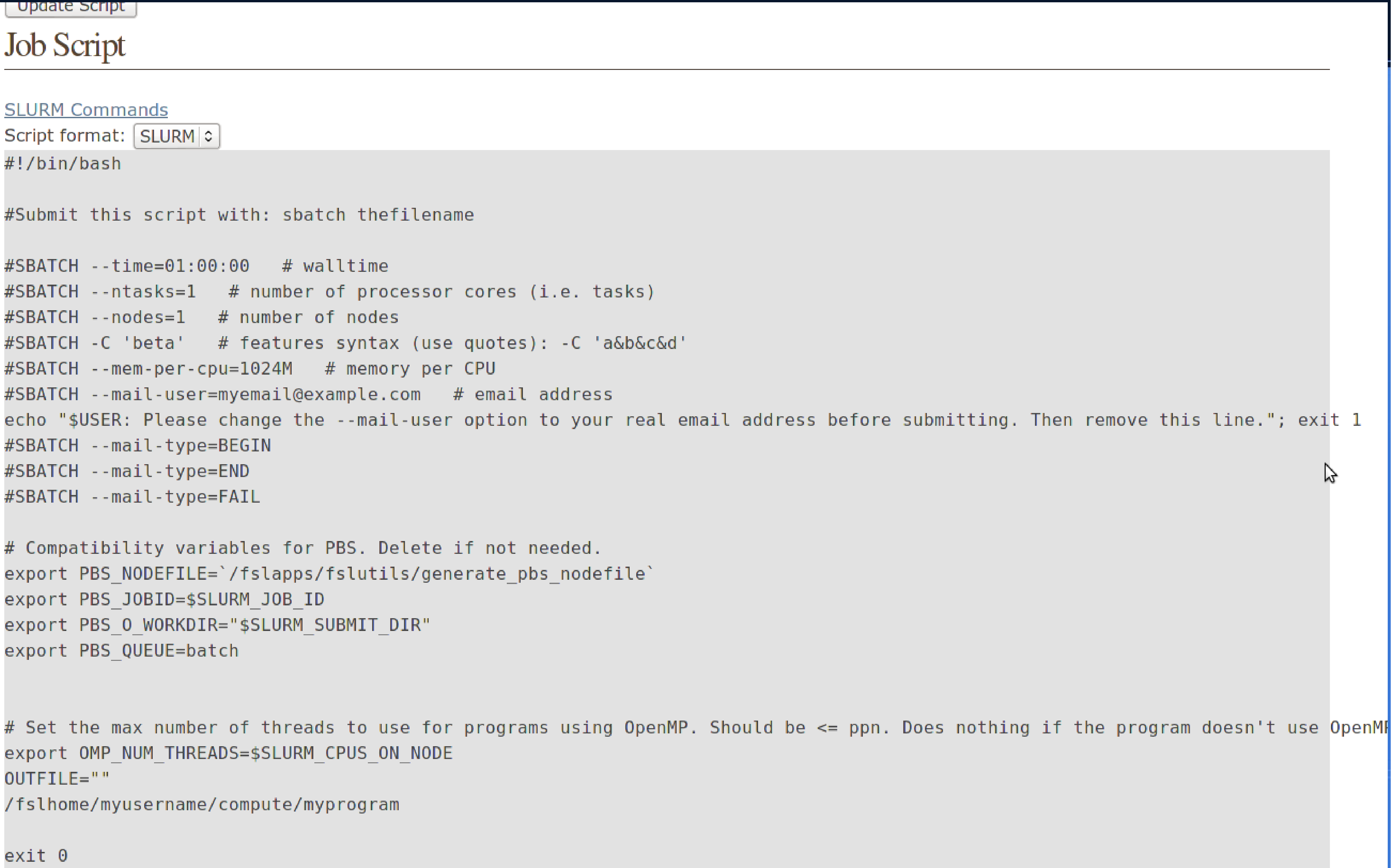
Features

If you don't know what these mean, you probably don't need to check them. The more you check, the fewer nodes you can run on. More information

If you must guarantee that your jobs use specific hardware (e.g. for benchmarking) please contact FSL.

□**amd** [?] Nodes avail: 1/2 Procs avail: 24/32	□**avx** [?] Nodes avail: 1/320 Procs avail: 1244/5120	☑**beta** [?] Nodes avail: 33/683 Procs avail: 3357/9556	□**gpu** [?] Nodes avail: 1/3 Procs avail: 12/28
□**ib** [?]	□**intel** [?]	□**m2050** [?]	□**s1070** [?]

Script Generator (2 of 2)



Job Script

SLURM Commands

Script format: SLURM

```
#!/bin/bash

#Submit this script with: sbatch thefilename

#SBATCH --time=01:00:00   # walltime
#SBATCH --ntasks=1   # number of processor cores (i.e. tasks)
#SBATCH --nodes=1   # number of nodes
#SBATCH -C 'beta'   # features syntax (use quotes): -C 'a&b&c&d'
#SBATCH --mem-per-cpu=1024M   # memory per CPU
#SBATCH --mail-user=myemail@example.com   # email address
echo "$USER: Please change the --mail-user option to your real email address before submitting. Then remove this line."; exit 1
#SBATCH --mail-type=BEGIN
#SBATCH --mail-type=END
#SBATCH --mail-type=FAIL

# Compatibility variables for PBS. Delete if not needed.
export PBS_NODEFILE=`/fslapps/fslutils/generate_pbs_nodefile`
export PBS_JOBID=$SLURM_JOB_ID
export PBS_O_WORKDIR="$SLURM_SUBMIT_DIR"
export PBS_QUEUE=batch



# Set the max number of threads to use for programs using OpenMP. Should be <= ppn. Does nothing if the program doesn't use OpenMP
export OMP_NUM_THREADS=$SLURM_CPUS_ON_NODE
OUTFILE=""
/fslhome/myusername/compute/myprogram

exit 0
```

Wishlist (1 of 2)

- ~~Custom job submit plugin error messages~~ (in 13.12)
- Only *n* jobs per user or account accrue queue time for priority calculation purposes (eliminate benefits of queue stuffing)
- Include accrued CPU time of running jobs in fairshare calculations
  - Currently, infrequent users can flood the system with jobs until some of the jobs finish
- Transient failure handler like Torque pbs_mom's ERROR: messages (we use reservations instead)

Wishlist (2 of 2)

- Per node per job stats
  - Memory and CPU efficiency (used / allocated)
- cgroup enhancement: catch processes launched through ssh
  - Create cgroups on each allocated node for a job even if the node has no job steps (conf option?)
  - Use /etc/ssh/sshrc to assign to job cgroup
  - ssh{,d}_config: AcceptEnv/SendEnv SLURM_JOB_ID
  - Finish jobacct_gather/cgroup plugin (13.12?)
  - New option? “scontrol cgroup addpid jobid=<jobid> pid=<pid>”

Questions?