Using Balena

Bath University has a mid-size cluster named Balena
Using Balena

Bath University has a mid-size cluster named Balena

It has

• 196 nodes of a pair of Intel Xeon E5-2650v2 Ivybridge cpus, 2.6GHz, each having 8 cores, thus 16 symmetric shared memory cores on a node

• 17 nodes of a pair of Intel Xeon Gold 6126 Skylake cpus, 2.6GHz, each having 12 cores, thus 24 symmetric shared memory cores on a node

This gives us 3544 cpu cores
Using Balena

Bath University has a mid-size cluster named Balena

It has

- 196 nodes of a pair of Intel Xeon E5-2650v2 Ivybridge cpus, 2.6GHz, each having 8 cores, thus 16 symmetric shared memory cores on a node
Using Balena

Bath University has a mid-size cluster named Balena

It has

- 196 nodes of a pair of Intel Xeon E5-2650v2 Ivybridge cpus, 2.6GHz, each having 8 cores, thus 16 symmetric shared memory cores on a node
- 17 nodes of a pair of Intel Xeon Gold 6126 Skylake cpus, 2.6GHz, each having 12 cores, thus 24 symmetric shared memory cores on a node
Using Balena

Bath University has a mid-size cluster named Balena

It has

- 196 nodes of a pair of Intel Xeon E5-2650v2 Ivybridge cpus, 2.6GHz, each having 8 cores, thus 16 symmetric shared memory cores on a node
- 17 nodes of a pair of Intel Xeon Gold 6126 Skylake cpus, 2.6GHz, each having 12 cores, thus 24 symmetric shared memory cores on a node

This gives us 3544 cpu cores
Using Balena

Most nodes have 64GB each, shared between the cores on the node.
Using Balena

Most nodes have 64GB each, shared between the cores on the node

Some have 128GB or 192GB, and for large computations, two have 512GB
Using Balena

There are 22 Nvidia K20x GPUs and 7 Nvidia P100 GPUs acting as coprocessors to the CPUs in various configurations.
Using Balena

There are 22 Nvidia K20x GPUs and 7 Nvidia P100 GPUs acting as coprocessors to the CPUs in various configurations.

A K20x GPU card has 2688 cores and a P100 has 3584 cores, but you can’t simply compare GPU cores with CPU cores numerically!
Using Balena

- 1 node with 2 AMD S10000 GPU cards
Using Balena

- 1 node with 2 AMD S10000 GPU cards

A S10000 GPU card has two 1792-core GPUs, totalling
\[2 \times 2 \times 1792 = 7168\] cores
Using Balena

- 8 nodes with 1 Intel Xeon Phi 5110P

A Xeon Phi 5110P has 60 (shared memory) cores, thus
providing $(8 + 3 \times 4) \times 60 = 1200$ cores
Using Balena

- 8 nodes with 1 Intel Xeon Phi 5110P
- 3 nodes with 4 Intel Xeon Phi 5110P
Using Balena

- 8 nodes with 1 Intel Xeon Phi 5110P
- 3 nodes with 4 Intel Xeon Phi 5110P

A Xeon Phi 5110P has 60 (shared memory) cores, thus providing $(8 + 3 \times 4) \times 60 = 1200$ cores
Using Balena

Plus more nodes for visualisation and interactive development
Using Balena

The nodes are connected by Infiniband QDR (40Gb/s aggregate)
Using Balena

The nodes are connected by Infiniband QDR (40Gb/s aggregate)

10 Gigabit Ethernet: bandwidth < 10Gb/s; latency 100µs

QDR Infiniband: bandwidth 8Gb/s; latency < 1.3µs
Using Balena

They run the Linux operating system
Using Balena

They run the Linux operating system

Actually a version of Redhat
Using Balena

They run the Linux operating system

Actually a version of Redhat

Modified by Scientific Linux
Using Balena

They run the Linux operating system
Actually a version of Redhat
Modified by Scientific Linux
Modified by Cluster Vision
Using Balena

It was commissioned to support running large parallel programs, and is not primarily a teaching resource.
Using Balena

It was commissioned to support running large parallel programs, and is not primarily a teaching resource.

It is a shared resource, so there are controls on how the nodes are allocated and used.
Using Balena

It was commissioned to support running large parallel programs, and is not primarily a teaching resource.

It is a shared resource, so there are controls on how the nodes are allocated and used.

In particular, you can’t just use any bunch of nodes you feel like: access is mediated by a batch submission system.
Using Balena

It was commissioned to support running large parallel programs, and is not primarily a teaching resource.

It is a shared resource, so there are controls on how the nodes are allocated and used.

In particular, you can’t just use any bunch of nodes you feel like: access is mediated by a *batch submission* system.

Welcome to the 1960s!
Using Balena

It was commissioned to support running large parallel programs, and is not primarily a teaching resource.

It is a shared resource, so there are controls on how the nodes are allocated and used.

In particular, you can’t just use any bunch of nodes you feel like: access is mediated by a batch submission system.

Welcome to the 1960s!

The system is named SLURM (“Simple Linux Utility for Resource Management”)
Using Balena

It was commissioned to support running large parallel programs, and is not primarily a teaching resource.

It is a shared resource, so there are controls on how the nodes are allocated and used.

In particular, you can’t just use any bunch of nodes you feel like: access is mediated by a batch submission system.

Welcome to the 1960s!

The system is named SLURM (“Simple Linux Utility for Resource Management”)

It is not simple!
Using Balena

To run a program you

• write and debug your program
• write a job submission script
• submit your program to the cluster
• wait for the results
Using Balena

To run a program you

• write and debug your program
Using Balena

To run a program you

- write and debug your program
- write a job submission script
Using Balena

To run a program you

- write and debug your program
- write a job submission script
- submit your program to the cluster
Using Balena

To run a program you

- write and debug your program
- write a job submission script
- submit your program to the cluster
- wait for the results
Using Balena

Normally small jobs have a fast (sometimes immediate) turnaround, so it’s not as if you have to wait a week for your results
Using Balena

Normally small jobs have a fast (sometimes immediate) turnaround, so it’s not as if you have to wait a week for your results.

BUCS have a lot of Web pages describing the usage of Balena: you should have a browse through them.
Using Balena

Normally small jobs have a fast (sometimes immediate) turnaround, so it’s not as if you have to wait a week for your results.

BUCS have a lot of Web pages describing the usage of Balena: you should have a browse through them.

We shall only give an outline here.
To use Balena you need to log into a *head node*.
Using Balena

Use `ssh balena.hpc.bath.ac.uk` to login to a head node: there is no direct access to the nodes in the cluster.
Using Balena

Use `ssh balena.hpc.bath.ac.uk` to login to a head node: there is no direct access to the nodes in the cluster.

From Windows, you can use Kitty.
Using Balena

Use `ssh balena.hpc.bath.ac.uk` to login to a head node: there is no direct access to the nodes in the cluster.

From Windows, you can use Kitty.

Note that, for security, you can only access Balena from within the University of Bath.
Using Balena

Use `ssh balena.hpc.bath.ac.uk` to login to a head node: there is no direct access to the nodes in the cluster.

From Windows, you can use Kitty.

Note that, for security, you can only access Balena from within the University of Bath.

If you are off-site, you can either `ssh` to `linux.bath.ac.uk` and from there to Balena; or use the Uni’s VPN (https://www.bath.ac.uk/guides/setting-up-vpn-on-your-device/)
Using Balena

Balena has disk space separate from your other BUCCS space
Using Balena

Balena has disk space separate from your other BUCS space
This disk space is not visible outside the cluster
Using Balena

Balena has disk space separate from your other BUCCS space

This disk space is not visible outside the cluster

Your usual BUCCS filesystem is also mounted on the head node (but not the compute nodes) at a mountpoint given by $BUCSHOME
Using Balena

Balena has disk space separate from your other BUCS space.

This disk space is not visible outside the cluster.

Your usual BUCS file space is also mounted on the head node (but not the compute nodes) at a mountpoint given by $BUCSHOME.

So you can copy data back and forth simply as:

cp $BUCSHOME/path/to/prog.c dir/on/balena

on Balena.
Using Balena

But it’s easiest to keep your programs and data on Balena: write, compile and debug your program on a head node, copy the results back to main BUCS when you need to
But it’s easiest to keep your programs and data on Balena: write, compile and debug your program on a head node, copy the results back to main BUCLS when you need to.

However, the speed of access to Balena $HOME disk is deliberately kept slow as it is not intended for heavy use.
Using Balena

Instead, there is very fast scratch disk using the BeeGFS Parallel File System, accessible at $SCRATCH$, and through a soft link in your $HOME$ named scratch.

$ ls -l
total 0
lrwxrwxrwx 1 masrjb balena_cm 29 Sep 3 15:54 scratch -> /beegfs/scratch/user/u/masrjb

This currently is 0.6PB
Using Balena

Use scratch (big and fast) for reading and writing data while running your programs.
Using Balena

Use `scratch` (big and fast) for reading and writing data while running your programs

But take care: it is not backed up, it is shared, and contents may be deleted without warning
Using Balena

Use `scratch` (big and fast) for reading and writing data while running your programs

But take care: it is not backed up, it is shared, and contents may be deleted without warning

Use Balena `~/.amap` (small and slow) for things you need to keep
Using Balena

Use *scratch* (big and fast) for reading and writing data while running your programs

But take care: it is not backed up, it is shared, and contents may be deleted without warning

Use Balena `$HOME` (small and slow) for things you need to keep

And `$BUCSHOME` for getting stuff off the cluster
Using Balena

The head node is not designed for heavy use and it would be frowned upon if you tried to run big programs on it.
Using Balena

The head node is not designed for heavy use and it would be frowned upon if you tried to run big programs on it.

Compiling and editing stuff is OK, but not much more than that.
Using Balena

The head node is not designed for heavy use and it would be frowned upon if you tried to run big programs on it.

Compiling and editing stuff is OK, but not much more than that.

Big jobs should be run on the cluster, of course.
There is a lot of software available to run on the cluster, including several variants of compilers and parallel libraries.
Using Balena

There is a lot of software available to run on the cluster, including several variants of compilers and parallel libraries.

There is a simple *module* system that aids in selecting the right combination of bits of software, e.g., the GCC compiler and a version of MPI that is compatible with it.
Using Balena

There is a lot of software available to run on the cluster, including several variants of compilers and parallel libraries.

There is a simple *module* system that aids in selecting the right combination of bits of software, e.g., the GCC compiler and a version of MPI that is compatible with it.

Use `module avail` to see the list of available modules.
Using Balena

There is a lot of software available to run on the cluster, including several variants of compilers and parallel libraries.

There is a simple *module* system that aids in selecting the right combination of bits of software, e.g., the GCC compiler and a version of MPI that is compatible with it.

Use `module avail` to see the list of available modules.

Use `module load` to load a module.
Using Balena

You will need:

module load gcc
module load slurm
Using Balena

You will need:
module load gcc
module load slurm

Or just put those commands in your .bashrc to have them loaded every time you login
Using Balena

You will need:
module load gcc
module load slurm

Or just put those commands in your .bashrc to have them loaded every time you login

For Coursework 2 you will also want
module load openmpi/gcc
Using Balena

- `module list` lists the currently loaded modules;
- `module unload` to unload a module;
- `module purge` to unload all modules
Using Balena

When you have a working program, you need to submit a batch job for the cluster to run it
Using Balena

When you have a working program, you need to submit a batch job for the cluster to run it

This entails writing a batch submission script
A simple single processor job. Lines starting with `#SBATCH` are options for the `sbatch` command:

```bash
#!/bin/sh
# Account & partition (must have these)
#SBATCH --account=cm30225
#SBATCH --partition=teaching
# Name of job (optional)
#SBATCH --job-name=Test_Serial
# one node
#SBATCH --nodes=1
# any normal shell stuff
pwd

# Run the program
./helloworld
```
Using Balena

This is a shell script, a plain text file, called, e.g., runhello.batch or runhello.slm
Using Balena

This is a shell script, a plain text file, called, e.g., runhello.batch or runhello.slm

Submit the job:
sbatch runhello.batch
Using Balena

This is a shell script, a plain text file, called, e.g., runhello.batch or runhello.slm

Submit the job:
sbatch runhello.batch

This will be submitted to a *run queue*, where it will sit until resources become available to run the job
Using Balena

This is a shell script, a plain text file, called, e.g., runhello.batch or runhello.slm

Submit the job:
sbatch runhello.batch

This will be submitted to a run queue, where it will sit until resources become available to run the job

When it runs, output to stdout will end up in file slurm-<jobnumber>.out and stderr in slurm-<jobnumber>.err
Using Balena

There are several queues, the cm30225 queue is the one which you will be using
Using Balena

There are several queues, the cm30225 queue is the one which you will be using.

The partition indicates a subset of the nodes: this unit has a partition of 7 nodes reserved for our exclusive use.
Using Balena

SLURM options:

- `#SBATCH --time=hh:mm:ss`
  Limit the amount of time the program can take. The program will be killed automatically if the time given is exceeded. `time` is real elapsed time.

You can set this to something reasonable (and small) to help SLURM determine how to schedule your job.
Using Balena

SLURM options:

- `#SBATCH --time=hh:mm:ss`
  Limit the amount of time the program can take. The program will be killed automatically if the time given is exceeded. `time` is real elapsed time.

You can set this to something reasonable (and small) to help SLURM determine how to schedule your job.

Setting this to the smallest value you need may encourage SLURM to run your job sooner.
Using Balena

SLURM options:

- `#SBATCH --time=hh:mm:ss`
  Limit the amount of time the program can take. The program will be killed automatically if the time given is exceeded. `time` is real elapsed time.

You can set this to something reasonable (and small) to help SLURM determine how to schedule your job.

Setting this to the smallest value you need may encourage SLURM to run your job sooner.

CM30225 has a default of 10 minutes, and a maximum of 15 minutes.
Using Balena

- `#SBATCH --mail-type=[events]`
  If you want to be notified when a job has finished, SLURM can send you an email. Use `END` to request an email for normal exit; `BEGIN` for when the job starts; `FAIL` for when the job fails. Mostly used for long running jobs.
Using Balena

- `#SBATCH --mail-type=[events]`
  If you want to be notified when a job has finished, SLURM can send you an email. Use `END` to request an email for normal exit; `BEGIN` for when the job starts; `FAIL` for when the job fails. Mostly used for long running jobs
- `#SBATCH --mail-user=[user]` to specify an email address
Using Balena

- `#SBATCH --job-name=[jobname]`
  A name for this job, mostly for human benefit. Also gives the default file names for the where the output from your program goes (see above)
Using Balena

- `#SBATCH --job-name=[jobname]`
  A name for this job, mostly for human benefit. Also gives the default file names for the where the output from your program goes (see above)

- `#SBATCH --output=[filename]`
  Send program output to a named file
Using Balena

- `#SBATCH --job-name=[jobname]`
  A name for this job, mostly for human benefit. Also gives the default file names for the where the output from your program goes (see above)
- `#SBATCH --output=[filename]`
  Send program output to a named file
- `#SBATCH --error=[filename]`
  Send program error output to a named file
Using Balena

- `#SBATCH --nodes=[n]`
  Specify the number of nodes your job wants
Using Balena

- `#SBATCH --nodes=[n]`
  Specify the number of nodes your job wants

- `#SBATCH --ntasks-per-node=[n]`
  Specify the number of MPI processes on a node (see later)
Using Balena

- `#SBATCH --nodes=[n]`
  Specify the number of nodes your job wants
- `#SBATCH --ntasks-per-node=[n]`
  Specify the number of MPI processes on a node (see later)

And more to control maximum memory used, etc.
Using Balena

- `#SBATCH --nodes=[n]`
  Specify the number of nodes your job wants

- `#SBATCH --ntasks-per-node=[n]`
  Specify the number of MPI processes on a node (see later)

And more to control maximum memory used, etc.

The teaching partition allows you to use up to a maximum of 4 nodes at a time, so up to 64 distributed memory cpu cores (coursework 2)
Using Balena

The maximum number of cores in a shared memory configuration is 16 (coursework 1)
Using Balena

The maximum number of cores in a shared memory configuration is 16 (coursework 1)

It is pointless asking for more than 16 cores on a node and SLURM will reject such a request
Using Balena

Note on `ntasks-per-node`: you will be allocated exclusive use of whole nodes, so you will always have access to 16 cores on a node regardless of this value.
Note on `ntasks-per-node`: you will be allocated exclusive use of whole nodes, so you will always have access to 16 cores on a node regardless of this value.

So in a single-node, shared memory program its value is effectively ignored.
Note on `ntasks-per-node`: you will be allocated exclusive use of whole nodes, so you will always have access to 16 cores on a node regardless of this value.

So in a single-node, shared memory program its value is effectively ignored.

But when running MPI the values of `nodes` and `ntasks-per-node` are needed and used in the initialisation and connection of the separate MPI processes.
To repeat: jobs are given exclusive use of nodes and no other user’s jobs will be scheduled on your cores while your job is running
To repeat: jobs are given exclusive use of nodes and no other user’s jobs will be scheduled on your cores while your job is running.

There may be a few system processes, but otherwise the allocated cores are occupied 100% running your job.
To repeat: jobs are given exclusive use of nodes and no other user’s jobs will be scheduled on your cores while your job is running.

There may be a few system processes, but otherwise the allocated cores are occupied 100% running your job.

If your program is running slowly, it’s not because someone else is sharing your cores!
Using Balena

When you submit a job
sbatch jobfile
will reply with a job id unique to that job
Using Balena

When you submit a job
`sbatch jobfile`
will reply with a `job id` unique to that job

To check on the progress of a job use
`squeue`

or
`squeue -u username`

or
`squeue -p teaching`
for all jobs on the `teaching` partition
Using Balena

When you submit a job

sbatch jobfile
will reply with a job id unique to that job

To check on the progress of a job use

squeue
or
squeue -u username
or
squeue -p teaching
for all jobs on the teaching partition

To kill a job use

scancel jobid
where you can get jobid from squeue or from the original sbatch
Using Balena

More detailed information about a particular job:
scontrol show job jobid
Using Balena

More detailed information about a particular job:
\texttt{scontrol show job jobid}

More detailed information about a particular node:
\texttt{scontrol show node nodename}
Using Balena

More detailed information about a particular job:
scontrol show job \textit{jobid}

More detailed information about a particular node:
scontrol show node \textit{nodenname}

Use \texttt{sinfo} and \texttt{sview} for general status information
Using Balena

More detailed information about a particular job:
scontrol show job jobid

More detailed information about a particular node:
scontrol show node nodename

Use sinfo and sview for general status information

sshare to see which accounts you have access to, and how much you have used them
Using Balena

More detailed information about a particular job:
scontrol show job jobid

More detailed information about a particular node:
scontrol show node nodename

Use sinfo and sview for general status information

sshare to see which accounts you have access to, and how much you have used them

sprio -u username to see the priority of your jobs
Using Balena

More detailed information about a particular job:
scontrol show job jobid

More detailed information about a particular node:
scontrol show node nodename

Use sinfo and sview for general status information

sshare to see which accounts you have access to, and how much you have used them

sprio -u username to see the priority of your jobs

And more, like suspending a job, moving jobs between queues and so on
If you submit a large number of jobs your priority will decrease to give other people's jobs a chance.
Using Balena

If you submit a large number of jobs your priority will decrease to give other people’s jobs a chance

Remember that the teaching partition is shared amongst the entire class of CM30225
Using Balena

If you submit a large number of jobs your priority will decrease to give other people’s jobs a chance.

Remember that the teaching partition is shared amongst the entire class of CM30225.

Expect the queue to get longer as you approach hand-in date!
Using Balena

Using MPI

MPI is supported on Balena
Using Balena
Using MPI

MPI is supported on Balena

Follow the instructions on
http://wiki.bath.ac.uk/display/HPC/OpenMPI
Using Balena

Using MPI

MPI is supported on Balena

Follow the instructions on http://wiki.bath.ac.uk/display/HPC/OpenMPI

There are several versions of MPI on Balena: we need to pick one:
module load openmpi/gcc
Using Balena

Using MPI

MPI is supported on Balena

Follow the instructions on http://wiki.bath.ac.uk/display/HPC/OpenMPI

There are several versions of MPI on Balena: we need to pick one:

module load openmpi/gcc

OpenMPI seems the simplest to use, but there are others if you feel brave
Using Balena

Using MPI

MPI is supported on Balena

Follow the instructions on http://wiki.bath.ac.uk/display/HPC/OpenMPI

There are several versions of MPI on Balena: we need to pick one:
module load openmpi/gcc

OpenMPI seems the simplest to use, but there are others if you feel brave

Put the load above in your .bashrc so you can forget about it
Compile your code using `mpicc`
Using Balena
Using MPI

Compile your code using `mpicc`

```bash
mpicc -Wall -Wextra -Wconversion -o prog prog.c
```
Using Balena

Using MPI

Compile your code using `mpicc`

`mpicc -Wall -Wextra -Wconversion -o prog prog.c`

This links in the necessary MPI libraries
Using Balena
Using MPI

Compile your code using `mpicc`

```sh
mpicc -Wall -Wextra -Wconversion -o prog prog.c
```

This links in the necessary MPI libraries

The modules loaded make sure the right libraries are linked in for your chosen version of MPI
The number of processors used is specified in the SLURM script by `nodes` and `ntasks_per_node`
The number of processors used is specified in the SLURM script by nodes and ntasks_per_node.

To start an MPI program use (in the SLURM script)

```
mpirun -np n ./prog
```
The number of processors used is specified in the SLURM script by `nodes` and `ntasks_per_node`.

To start an MPI program use (in the SLURM script)
```
mpirun -np n ./prog
```

The `np` must equal `nodes \times ntasks_per_node`
In fact, there is an environment variable set by SLURM:

```
SLURM_NTASKS
```
In fact, there is a environment variable set by SLURM: `SLURM_NTASKS`

This is computed by SLURM to be the right value for `np`:

```
mpirun -np $SLURM_NTASKS ./prog
```
In fact, there is an environment variable set by SLURM: `SLURM_NTASKS`

This is computed by SLURM to be the right value for `np`:
```
mpirun -np $SLURM_NTASKS ./prog
```

`mpirun` does all the hard work of setting up the processes on each processor and connecting them over the network or via shared memory, as appropriate.
mpirun also

- connects the standard I/O streams from the remote processes to the master process (rank 0), so, for example, a `printf` anywhere will appear in the output file
mpirun also

- connects the standard I/O streams from the remote processes to the master process (rank 0), so, for example, a `printf` anywhere will appear in the output file
- sets several environment variables e.g., `SLURM_SUBMIT_DIR`, which names the directory the SLURM script was submitted from
Using Balena

Using MPI

mpirun also

- connects the standard I/O streams from the remote processes to the master process (rank 0), so, for example, a printf anywhere will appear in the output file
- sets several environment variables e.g., SLURM_SUBMIT_DIR, which names the directory the SLURM script was submitted from
- kills off all the processes and tidies up at the end; also if there is an error
A variety of debugging tools are available, depending on your level of sophistication.
A variety of debugging tools are available, depending on your level of sophistication

- Using print statements in your program is more effective than you might think!
A variety of debugging tools are available, depending on your level of sophistication

- Using print statements in your program is more effective than you might think!
- gdb
Using Balena
Debugging

A variety of debugging tools are available, depending on your level of sophistication

- Using print statements in your program is more effective than you might think!
- gdb
- valgrind
A variety of debugging tools are available, depending on your level of sophistication

- Using print statements in your program is more effective than you might think!
- gdb
- valgrind
- You may be able to install OpenMPI and run `mpicc` and `mpirun` on your own PC
A variety of debugging tools are available, depending on your level of sophistication

- Using print statements in your program is more effective than you might think!
- gdb
- valgrind
- You may be able to install OpenMPI and run mpicc and mpirun on your own PC
- Various graphical profiling and tracing tools, e.g., Allinea. Find and read the documentation!
Using Balena

Summary

- Programs and data should be on the scratch filesystem so they are visible to the compute nodes

Note: beware there is documentation on the Wiki for the old cluster (Aquila) as well as for Balena. Make sure you are reading the right pages!
Using Balena

Summary

- Programs and data should be on the **scratch** filesystem so they are visible to the compute nodes
- For a shared memory program on one node, use `--nodes=1`

Note: beware there is documentation on the Wiki for the old cluster (Aquila) as well as for Balena. Make sure you are reading the right pages!
Using Balena

Summary

- Programs and data should be on the scratch filesystem so they are visible to the compute nodes
- For a shared memory program on one node, use --nodes=1
- For an MPI program, compile using mpicc and run using mpirun
Using Balena

Summary

- Programs and data should be on the scratch filesystem so they are visible to the compute nodes
- For a shared memory program on one node, use `--nodes=1`
- For an MPI program, compile using `mpicc` and run using `mpirun`
- There is a lot of documentation online: [https://wiki.bath.ac.uk/display/BalenaHPC/Balena+High+Performance+Computing+Service](https://wiki.bath.ac.uk/display/BalenaHPC/Balena+High+Performance+Computing+Service)

Note: beware there is documentation on the Wiki for the old cluster (Aquila) as well as for Balena. Make sure you are reading the right pages!
Using Balena

Summary

- Programs and data should be on the **scratch** filesystem so they are visible to the compute nodes
- For a shared memory program on one node, use `--nodes=1`
- For an MPI program, compile using `mpicc` and run using `mpirun`
- There is a lot of documentation online: https://wiki.bath.ac.uk/display/BalenaHPC/Balena+High+Performance+Computing+Service
- Note: beware there is documentation on the Wiki for the old cluster (Aquila) as well as for Balena. Make sure you are reading the right pages!
Using Balena

Summary

Balena is a shared resource, so be kind to your classmates!