How to deal with common Scheduling Problems

Troubleshooting, Make sure you gather data that may be lost

- Keep any error messages you received.
- What environment setting were set, which module have you loaded when you submitted the job?
- When did the problem occur?
- The state of a cluster changes, jobs start and finish, nodes fail and are repaired. What were the output of commands that you ran on the system that make you think there is a problem or that you think analyst should see.

Keep a record of any output with script command.

“script <filename to write>”

Do your work

“exit” to stop writing the script.

If you need to add more information to the file

Use “Script −a <filename>”

tip: Put a date and time in the file name

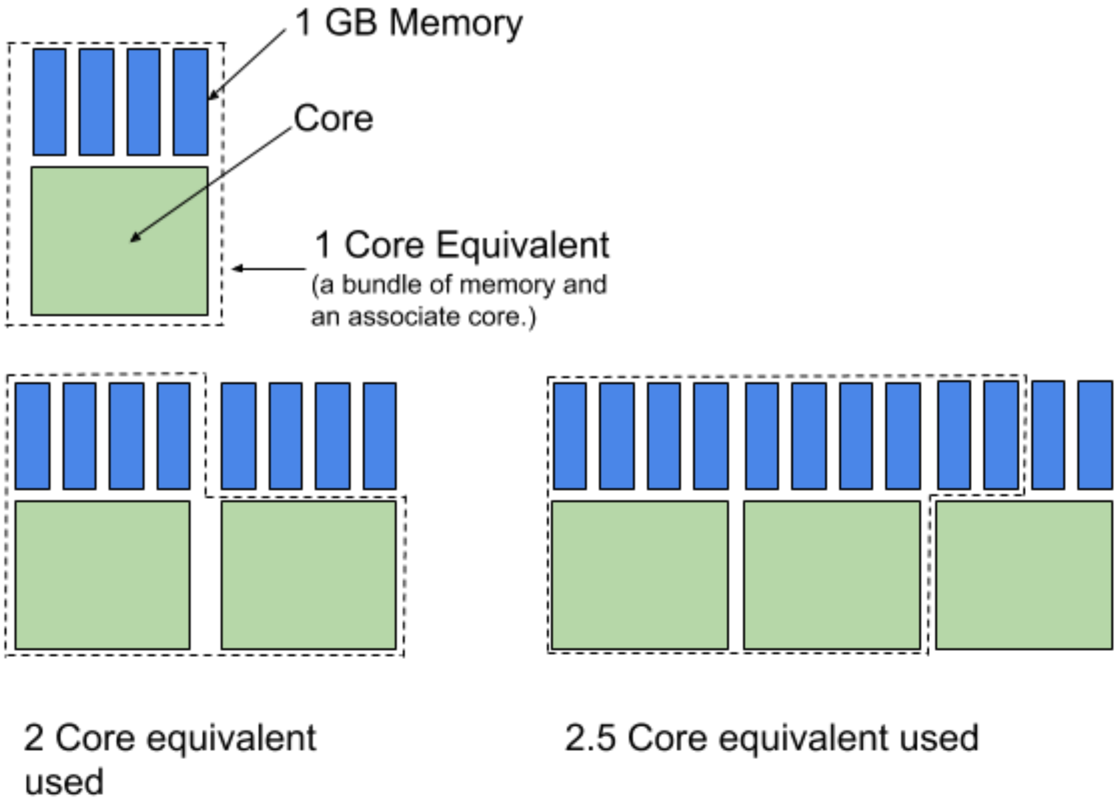
Memory Request Problems

- The Memory your jobs requests is what the system must have available for your job to use.
  - If your requests to use 1.5 TiB it will never run on a node with 1.5TiB of RAM as some RAM will be used by the operating system, and there will not be enough resources to run your job.
  - **Recommended** that all request for RAM in 1000’s of MiB
    - 1.5TiB=1,572,864 MiB and if you request 1,500,00 MiB there would be 72,864 MiB to run the OS and services.
    - This recommendation is also good for smaller single core jobs as well, On a 32 core 128 GiB RAM node, the scheduler can fit 31 jobs asking for 1 core and 4GiB of RAM or 32 jobs asking for 1 core an 4000MiB.

Running out of Memory

- Ask for more memory
- Be careful don’t ask for to much
    - Asking for more memory makes your job more difficult to schedule.
    - CC cluster only have 4GB RAM per core on most nodes.
    - You can ask for more memory than the cluster has
    - Your group will be assessed as having used more resources, (for the purpose priority and allocation)

Core Equivalent

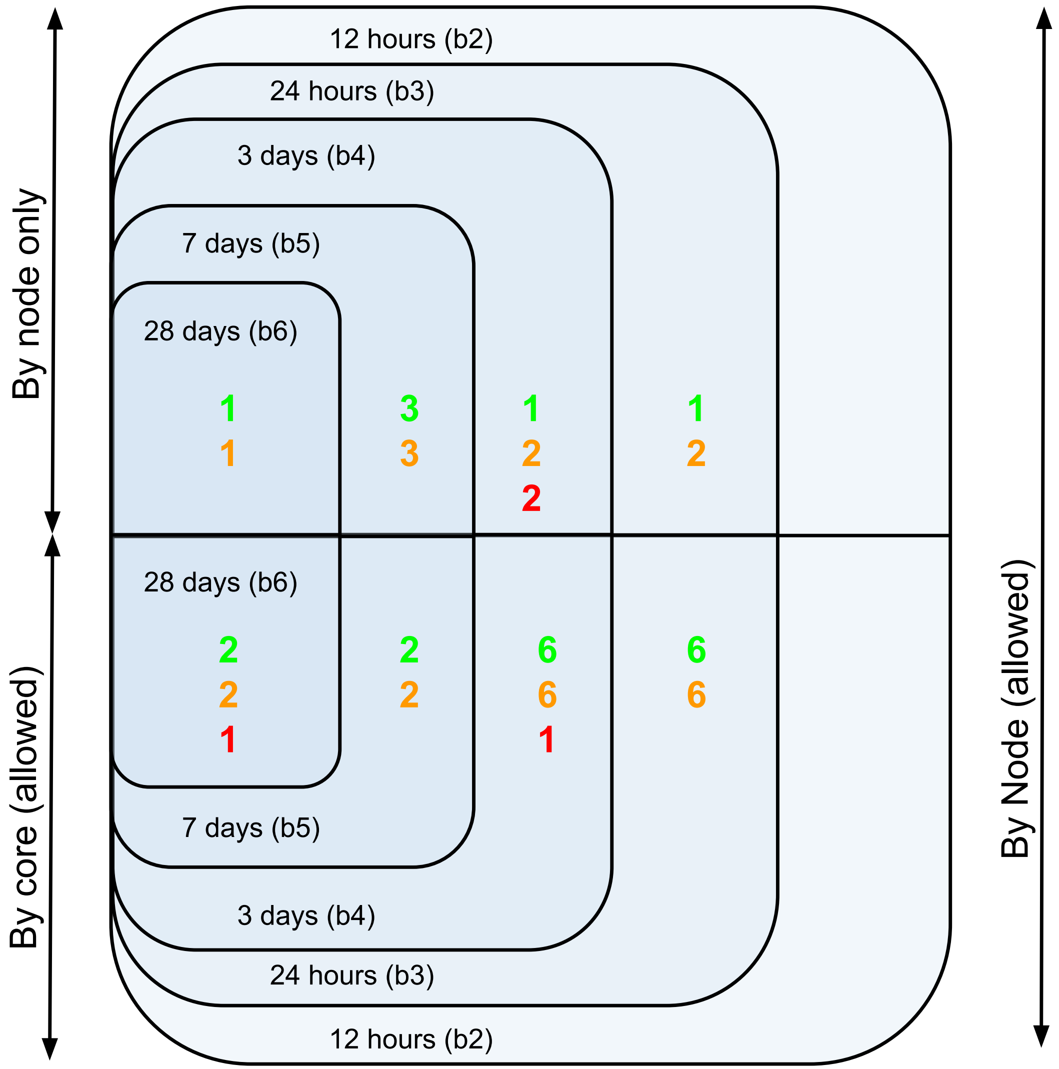


Compute Canada core equivalent documentation

Node types on Cedar

Number of Nodes	% of total	Memory per core (GiB)	Total Mem (GiB)	Cores	GPUS	Partition type
640	54.2%	4	192	48		cpubase
576	32.5%	4	128	32		
182	10.3%	8	256	32		
24	1.4%	16	512	32		cpularge
24	1.4%	48	1536	32		
4	0.2%	96	3072	32		
114	46.3%	4 (32 per GPU)	128	24	4	gpubase
132	53.7%	8 (64 per GPU)	256	24	4	gpularge

Sept 2018 Setup large mem partitions on Cedar



	28 days (b6)	7 days (b5)	3 days (b4)	24 hours (b3)	12 hours (b2)
By node only	🟩 1 🟧 1	🟩 3 🟧 3	🟩 1 🟧 2 🟥 2	🟩 1 🟧 2	
By core (allowed)	🟩 2 🟧 2 🟥 1	🟩 2 🟧 2	🟩 6 🟧 6 🟥 1	🟩 6 🟧 6	

🟩 512 GiB node 16GiB/core

🟧 1.5 GiB node 48GiB/core

🟥 3 TiB node 48GiB/core

Virtual and Physical memory

- Your program can ask to use a chunk of memory. This is virtual or requested MaxVMSize

- MaxRSS is the amount of memory used by your code.

Find out how much memory a job used.

Command	Flags	What its used for
sstat		Display various status information of a running job
	−j <jobid>	Displays information about the specified job
	--format= AveCPU,MaxRSS,MaxVMSize,JobID	limits the information to that about memory (MaxVMSize is requested memory) (MaxRSS is memory used)
sacct		Displays slurm accounting data
	−j <jobid>	Displays information about the specified job
	-u $USER	Displays information about jobs belong to a specific user
	--format= JobID,AveCPU,MaxRSS,MaxVMSize	limits the information to that about memory
salloc		Submit to run Job Interactively
	Same flags as sbatch	Note not all sbatch flags work

Interactive Jobs for debugging

Use salloc instead of sbatch to launch interactive jobs.

```
salloc  --ntasks=4 --mem-per-cpu 4000 -t 0-00:20
```

List environment variables with printenv

```
printenv | grep ^S
```

To check memory usage use

```
top –u $USER
```

Look at the jobs cgroup,

```
cat /sys/fs/cgroup/cpuset/slurm/uid_$SLURM_JOB_UID/
job_$SLURM_JOB_ID/step_$SLURM_STEPID/tasks
```

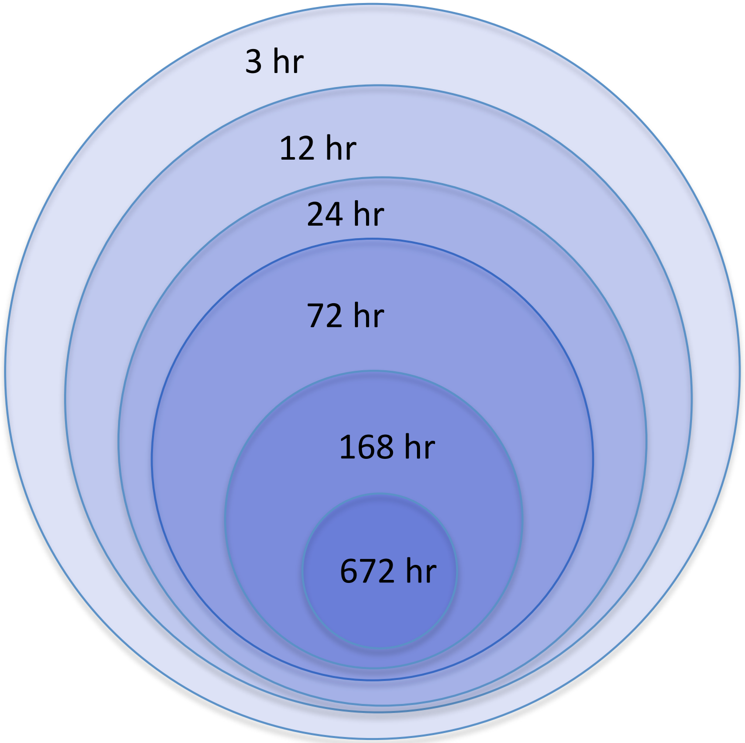
Remember that you are logged in only on one machine and your job may span more than one

Running out of runtime

If your jobs are running out of time.

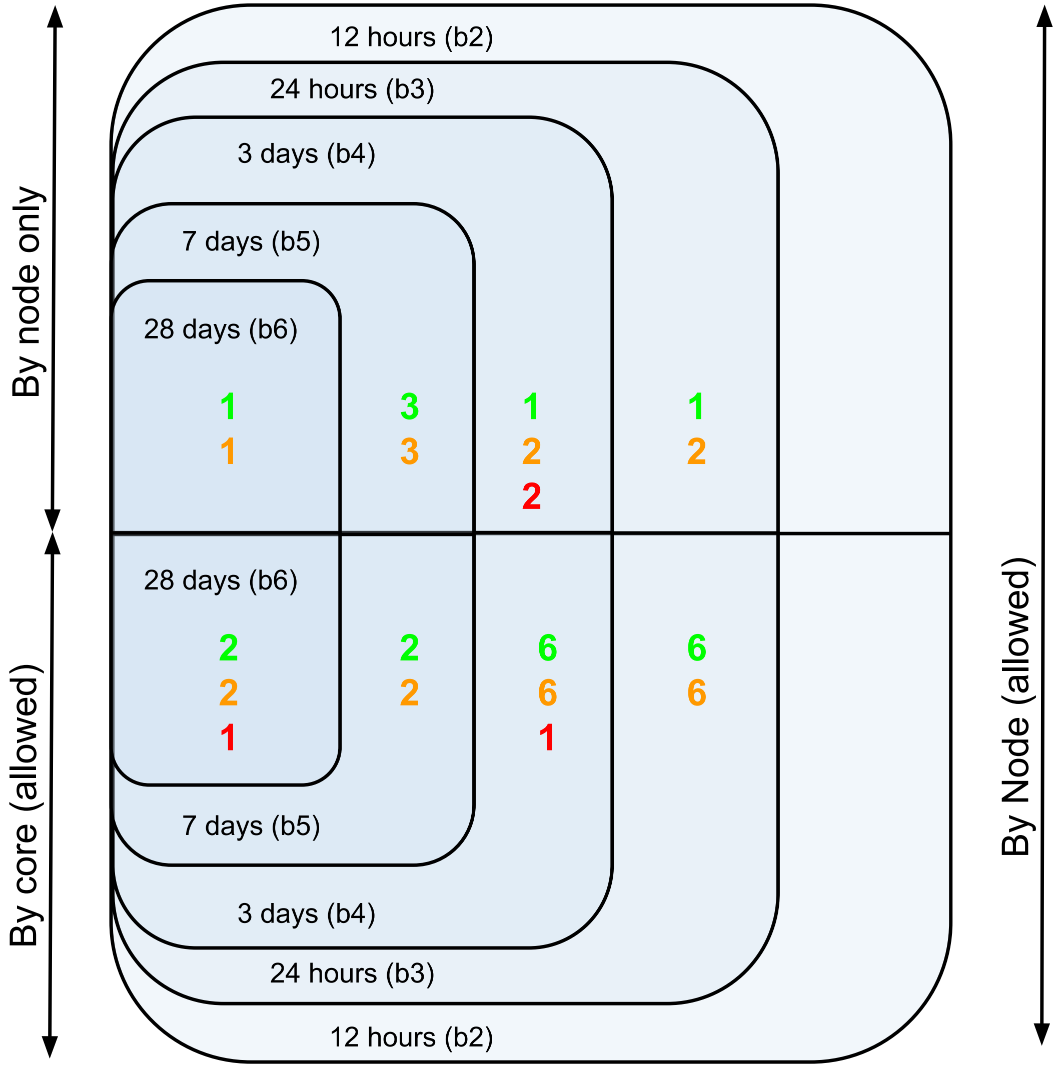
- Ask for more time.
- Don’t ask for to much runtime.
- Asking for more runtime may limit you to how many resources can run your job. This may interact with how much memory you asked for.

Partitions on Cedar and Graham



- There are partitions based upon how long the maximum walltime your job has.
- Your job ends up in the shortest walltime partition that has a longer walltime than your job
- The shorter walltime partitions include all the nodes of longer walltime partitions.

Sept 2018 Setup large mem partitions on Cedar



By node only

12 hours (b2)

24 hours (b3)

3 days (b4)

7 days (b5)

28 days (b6)

1
1

3
3

1
2
2

1
2

28 days (b6)

2
2
1

2
2

6
6
1

6
6

7 days (b5)

3 days (b4)

24 hours (b3)

12 hours (b2)

By core (allowed)

By Node (allowed)

512 GiB node 16GiB/core

1.5 GiB node 48GiB/core

3 TiB node 48GiB/core

Partition Stats

(CC script)

```
Node type |                     Max walltime
          |   3 hr   |  12 hr  |  24 hr  |  72 hr  |  168 hr |  672 hr |
----------|-------------------------------------------------------------
       Number of Queued Jobs by partition Type (by node:by core)
----------|-------------------------------------------------------------
Regular   |    1:15  |   2:31  |   2:145 |  11:187 |  86:69  |   3:2   |
Large Mem |    0:0   |   0:0   |   0:0   |   0:0   |   0:1   |   0:1   |
GPU       |    0:1   |   0:526 |  10:10  |   0:0   | 189:4   |   0:0   |
----------|-------------------------------------------------------------
      Number of Running Jobs by partition Type (by node:by core)
----------|-------------------------------------------------------------
Regular   |   60:6   |   4:2   |  45:836 |   5:90  |  11:1065|   1:4   |
Large Mem |    0:0   |   0:0   |   0:0   |   0:0   |   0:0   |   1:0   |
GPU       |    0:20  |   2:10  |  13:2   |   0:0   |   0:0   |   0:3   |
----------|-------------------------------------------------------------
        Number of Idle nodes by partition Type (by node:by core)
----------|-------------------------------------------------------------
Regular   |    0:0   |   0:0   |   0:0   |   0:0   |   0:0   |   0:0   |
Large Mem |    3:1   |   0:0   |   0:0   |   0:0   |   0:0   |   0:0   |
GPU       |   17:1   |  11:1   |   0:0   |   0:0   |   0:0   |   0:0   |
----------|-------------------------------------------------------------
       Total Number of nodes by partition Type (by node:by core)
----------|-------------------------------------------------------------
Regular   |  851:411 | 821:391 | 756:346 | 636:276 | 180:100 |  90:50  |
Large Mem |   27:12  |  24:11  |  24:11  |  20:3   |   3:2   |   2:1   |
GPU       |  156:78  | 144:72  | 116:58  | 104:52  |  13:12  |  13:12  |
----------|-------------------------------------------------------------
```

My job is not running right now

- `scontrol show job <jobid>`
  - pay special attention to **JobState**, and **Reason**
- Look at resource availability with CC's `partitions-stats` command, make sure you look up the resources the resources available to your jobs memory and time requirements.
- Looks at the jobs priority with the `sprio` command.

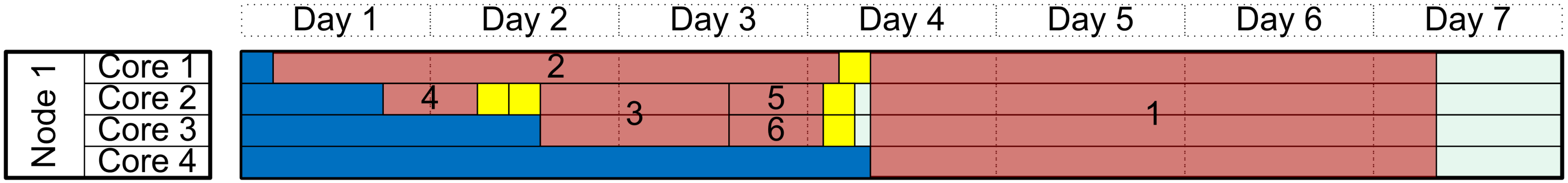
Fixing priority

- Priority is based upon your groups recent usage compared to allocation in the recent past.
  - Wait and ask group members to use less resources.
  - Ask your PI to apply for a RAC allocation.
  - Change your job to ask for resources that are in less demand. Ex: checkpointing.

Checkpointing

- Some programs can save its state and restart from save state.
- This allows long running jobs to be broken up into smaller runs.
- This minimizes effect of hardware failures, system downtime etc… on your ability to get your work done. This is particularly important for simulations using large amount of hardware.
- If your application does allow this make sure to understand how long a checkpoint and or restart takes. If it takes a long time to do, don’t do it very often.

Big high priority jobs in the scheduler leave “holes” that can be filled with smaller shorter Jobs



How to ask support for help

- Read status page and any support notices http://status.computecanada.ca/.
- Being able to ask for help in a helpful manner will likely result in your questions being help in a much more responsive manner.
- An email “Something is wrong”, “Nothing works” will take a long time to resolve.
  - A analyst responsible for a system will fix problems with the system they are responsible for, before they start answering asking you questions in order to determine which system you are having a problem. The problem could be with the cluster, cloud, website, network etc..
- In the subject of the email, include the system/cluster name a a few words of what may be wrong.
  - “job 123456 fails to run on the Cedar cluster” will much more likely to handled by a person who can help quickly.

Don’t make the Analysts play detective unnecessarily

- Compute Canada has multiple clusters.
- Your Compute Canada user name may not be apparent from your email.
- You may have 1000s of job queued, running, completed, failed on the system with which one did you have an issue?
- When did this happen?
- Which jobscript did you launch your job with, have you modified it since?
- What version of software are you running?

TO: support@computecanda.ca
Subject: Job 123456 gives errors on the CC Cedar cluster

Hello, My name is Alice, user asmith . Today at 10:00 am MST I submitted a job 123456 on the Cedar cluster. The Job script is located /my/job/script/path I have not changed it since submitting my job, since it is short I included it in the email bellow.

```
#!/bin/bash
#SBATCH --account=def-asmith-ab
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=16
#SBATCH --time=00:05:00
{ time mpiexec -n 1 ./sample1 ; } 2>out.time
```

A list of the following modules were loaded at the time follow:

```
[asmith@cedar5 ]$ module list
Currently Loaded Modules:
 1) nixpkgs/16.09  (S)    5) intel/2016.4  (t)
 2) icc/.2016.4.258 (H)   6) imkl/11.3.4.258 (math)
 3) gcccore/.5.4.0  (H)   7) openmpi/2.1.1 (m)
 4) ifort/.2016.4.258 (H)  8) StdEnv/2016.4 (S)
```

The job ran quickly and the myjob-123456.out and myjob-123456.err files were created.

There was no output in the myjob-123456.out file but there was an message in the myjob-123456.err output

```
[asmith@cedar5 scheduling]$ cat myjob-123456.err
slurmstepd: error: *** JOB 123456 ON cdr692 CANCELLED AT 2018-09-06T15:19:16 DUE TO TIME LIMIT ***
```

Can you tell me how to fix this problem?

User debugging questions

Any questions.

If there is time and we have a volunteer who consents for others to listen and learn and that there will be a recording made of this and posted on line. We can do a live debugging session example.

The end

Material bellow will not be in the presentation unless diagram is needed to answer a question.

SLURM SCRIPT REFERENCE MATERIAL BELLOW

Basic Slurm script commands

Slurm script command	Description
#!/bin/bash	Sets the shell that the job will be executed on the compute node
#SBATCH --ntasks=1 #SBATCH --n1	Requests for 1 processors on task, usually 1 cpu as 1 cpu per task is default.
#SBATCH --time=0-05:00 #SBATCH -t 0-05:00	Sets the maximum runtime of 5 hours for your job
#SBATCH --mail-user= <email>	Sets the email address for sending notifications about your job state.
#SBATCH --mail-type=BEGIN #SBATCH --mail-type=END #SBATCH --mail-type=FAIL #SBATCH --mail-type=REQUEUE #SBATCH --mail-type=ALL	Sets the scedualing system to send you email when the job enters the follwoing states: BEGIN,END,FAIL,REQUEUE,ALL
#SBATCH --job-name=my-named-job	Sets the Jobs name

Slurm script commands

Slurm script command	Description
#SBATCH −ntasks=X	Requests for X tasks. When cpus-per-task=1 (and this is the default) this requests X cores. When not otherwise constraint these CPUs may be running on any node
#SBATCH --nodes=X	Request that a minimum of X nodes be allocated to this job
#SBATCH --nodes=X-Y	Request that a minimum of X nodes and a maximum of Y nodes be allocated to this job
#SBATCH --cpus-per-task=X	Request that a minimum of X CPUs per task be allocated to this job
#SBATCH --tasks-per-node=X	Requests minimum of X task be allocated per node

Slurm script commands

Slurm script commands	Description of effects
#SBATCH --ntasks=1 #SBATCH --cpus-per-task=1	Requests 1 CPU (Serial) cpus-per-task is set to 1 by default and may be omitted.
#SBATCH --cpus-per-task=X #SBATCH --ntasks=1 #SBATCH --nodes=1	Requests for X CPUs in 1 task on 1 node (OpenMP) Both ntasks and nodes are set to 1 by default and may be omitted
#SBATCH --ntasks=X #SBATCH --tasks-per-node=X #SBATCH --cpus-per-task=1	Requests for X CPUs and tasks on 1 node cpus-per-task is set to 1 by default and may be omitted.
#SBATCH --ntasks=X #SBATCH --nodes=1 #SBATCH --cpus-per-task=1	Requests for X CPUs and tasks on 1 node cpus-per-task is set to 1 by default and may be omitted.

Slurm script commands

Slurm script commands	Description of effects
#SBATCH --ntasks=X #SBATCH --cpus-per-task=1	Requests X CPUs and tasks (MPI) cpus-per-task is set to 1 by default and may be omitted.
#SBATCH --ntasks=X #SBATCH --ntasks-per-node=Y #SBATCH --cpus-per-task=1	Requests for X CPUs and tasks with Y CPUs and tasks per node cpus-per-task is set to 1 by default and may be omitted.
#SBATCH --ntasks=X #SBATCH --nodes=1 #SBATCH --cpus-per-task=1	Requests for X CPUs and tasks on the same node, cpus-per-task is set to 1 by default and may be omitted.
#SBATCH --ntasks=X #SBATCH --nodes=1 #SBATCH --cpus-per-task=1	Requests for X CPUs and tasks on the 1 node cpus-per-task is set to 1 by default and may be omitted.

Slurm script commands

Slurm script command	Description
#SBATCH --ntasks=1 #SBATCH --cpus-per-task=1	Requests 1 cpu in 1 task. (Serial) cpus-per-task is set to 1 by default and may be omitted.
#SBATCH --cpus-per-task=N #SBATCH --ntasks=1 #SBATCH --nodes=1	Requests for X processors on the same node (OpenMP) Both ntasks and nodes are set to 1 by default and may be omitted
#SBATCH --ntasks=X	Requests for X processors which may be running on any node (MPI).
#SBATCH --nodes=X #SBATCH --ntasks=Y	Requests minimum of X nodes for the Y tasks. (MPI job)
#SBATCH --array=0-4	Requests Job array of 5 jobs with indexes 0-4
#SBATCH --array=1,3,5,7,9	Requests Job array of 5 jobs with indexes 1,3,5,7,9
#SBATCH --array=0-X%Y ex: #SBATCH --array=0-4%2	Requests Requests Job array of X jobs with only a maximum of Y jobs running at the same time

Slurm script commands

PBS script command	Description
#SBATCH --mem=4000	Requests 4000 MB of memory in total
#SBATCH --mem-per-cpu=4000	Requests 4000 MB of memory per cpu
#SBATCH --licenses=sas:2	Requests 2 SAS licenses
#SBATCH --gres=gpu:1	Requests that your job get 1 GPU allocated per node
#SBATCH --exclusive	Requests that your job run only on nodes with no other running jobs
#SBATCH --dependency=after:job_id1	Requests that the the job start after job (jobid1) has **started**
#SBATCH --dependency=afterany:job_id1, job_i2	Requests that the the job start after ether job (jobid1) or job (jobud2) has **finished**
#SBATCH --dependency=afterok:job_id1	Requests that the the job start after job (jobid1) has **finished successfully**

Slurm script commands

PBS script command	Description
#SBATCH --account=acc_name	To submit a job to a specific accounting group such as RAC/RAS allocation or different role
#SBATCH --tmp=200G	Asks for 200Gb of temporary disk space
#SBATCH --constraint=blue	To ask for a node feature or constraint set by cluster admin. Here we are looking for “blue” nodes.
#SBATCH --partition=partition_name	To ask for the job to run in a specific partition or queue by name, (unlike Moab there can be more than 1 partition per Job)
--prolog=<executable>	Run by srun only, runs the executable before the step
--epilog=<executable>	Run by srun only, runs the executable after the step finishes

SLURM Environment Variables

Environment Variable	Description
SLURM_JOB_NAME	User specified job name
SLURM_JOB_ID	Unique slurm job id
SLURM_NNODES	Number of nodes allocated to the job
SLURM_NTASKS	Number of tasks allocated to the job
SLURM_ARRAY_TASK_ID	Array index for this job
SLURM_ARRAY_TASK_MAX	Total number of array indexes for this job
SLURM_MEM_PER_CPU	Memory allocated per CPU
SLURM_JOB_NODELIST	List of nodes on which resources are allocated to Job
SLURM_JOB_CPUS_PER_NODE	Number of CPUs allocated per Node
SLURM_JOB_PARTITION	List of Partition(s) that the job is in.
SLURM_JOB_ACCOUNT	Account under which this job is run.

Getting information on your Job

Command	What its used for
squeue -u <username>	List all current jobs for a user
squeue -u <username> -t PENDING	List all pending jobs for a user
squeue -u <username> -t RUNNING	List all running jobs for a user
squeue -p <partitionname>	List all the jobs in a partition
scontrol show job <jobid>	List information on Job
scontrol show jobid -dd <jobid>	List detailed information on Job
sstat --format=AveCPU,MaxRSS,MaxVMSize,JobID -j <jobid>	List info resource used by your completed job : average cpu time, Max memory, Max virtual memory, JobId
sacct −u <username> --format=JobID,JobName,AveCPU,MaxRSS,MaxVMSize,JobID,Elapsed	List resources used by all jobs of a user
sprio	List job priority information

Controlling jobs

Command	What its used for
scancel <jobid>	Cancel job
scancel -u <username>	Cancel all the jobs for a user
scancel -t PENDING -u <username>	Cancel all the pending jobs for a user:
Scancel -name JobName	Cancel one or more jobs by name
scontrol hold <jobid>	Hold a job, prevent it form starting
scontrol resume <jobid>	Release a job hold, allowing the job to try to start
scontrol requeue <jobid>	Requeue a running, suspended or finished job into pending state
scontrol requeuehold<jobid>	First requeue the job than put a hold on it.
squeue -u <username> -ho %A -t R	List running jobs by user
squeue --start	Show expected start time of jobs. (This can change)

Getting information on you and your group

Command	What its used for
sacctmgr list Users USERS=<username>	List user and their default account (accounting group)
sacctmgr show user <username> withassoc	List user and their default account (accounting group) and shows more extensive information
sshare	Shows usage info for user.

Getting information on your Cluster

Command	What its used for
sinfo --states=idle	Show idle node on cluster
sinfo -R	Show down, drained and draining nodes and their reason
sinfo --Node --long	Show detailed node info.
scontrol show reservation	Shows reservations on the cluster

scontrol create reservation
user=root starttime=now
duration=infinite
flags=maint
nodes=<nodeid>	

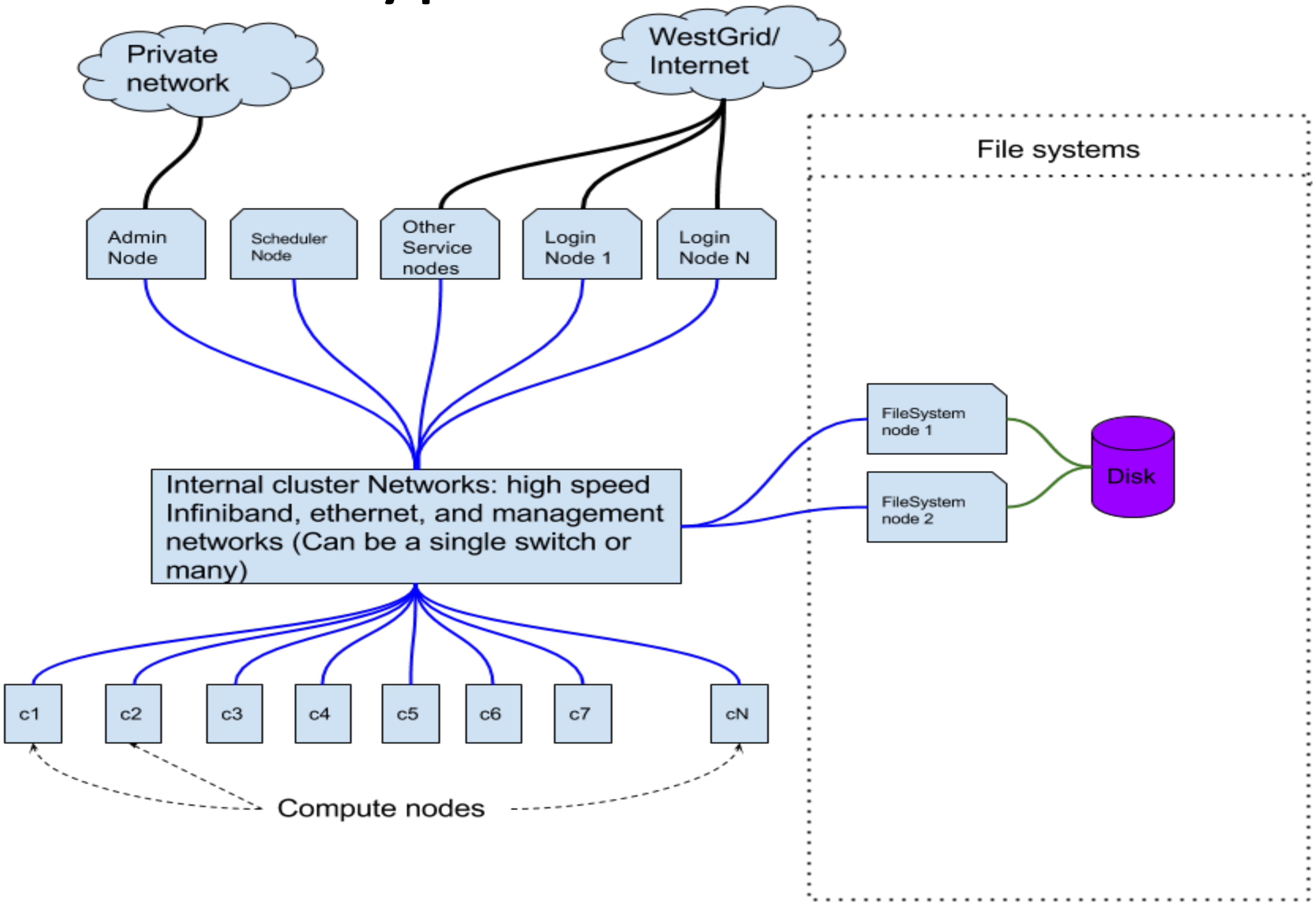
Administrating your Cluster

Command	What its used for
scontrol create reservation user=root starttime=now duration=infinite flags=maint nodes=<nodeid>	Create a maintaince reservation on node nodeid
sacctmgr modify account where name=def-<account> set rawusage=0	Zero account usage fairshare stats
sacctmgr modify user where account=def-<account> name=<uname> set RawUsage=0	Zero user usage fairshare stats

Material bellow will not be in the presentation unless diagram is needed to answer a question.

OTHER REFERENCE MATERIAL BELLOW

Typical HPC Cluster



Private network

WestGrid/ Internet

File systems

Admin Node

Scheduler Node

Other Service nodes

Login Node 1

Login Node N

FileSystem node 1

FileSystem node 2

Disk

Internal cluster Networks: high speed Infiniband, ethernet, and management networks (Can be a single switch or many)

c1

c2

c3

c4

c5

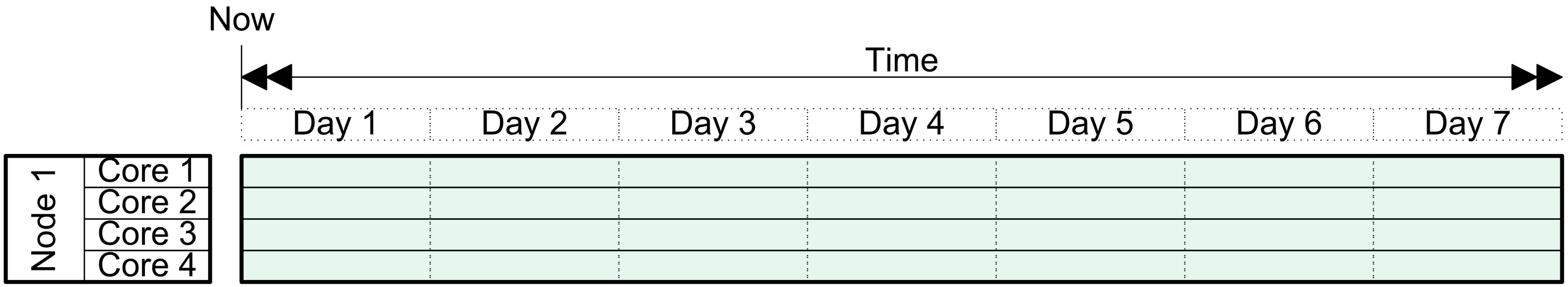
c6

c7

cN

Compute nodes

Visualizing single node cluster



Now

Time

Day 1

Day 2

Day 3

Day 4

Day 5

Day 6

Day 7

Node 1

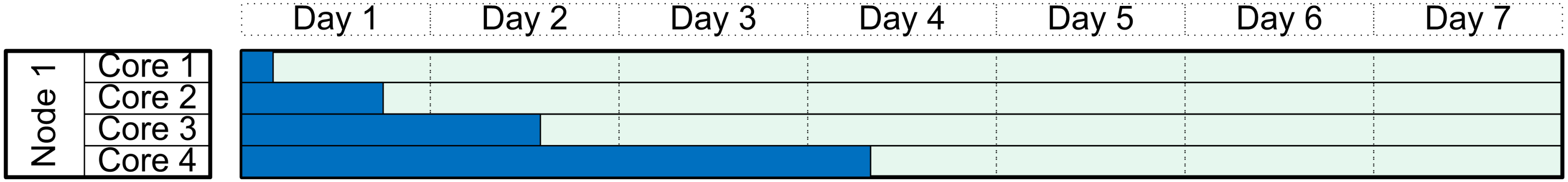
Core 1

Core 2

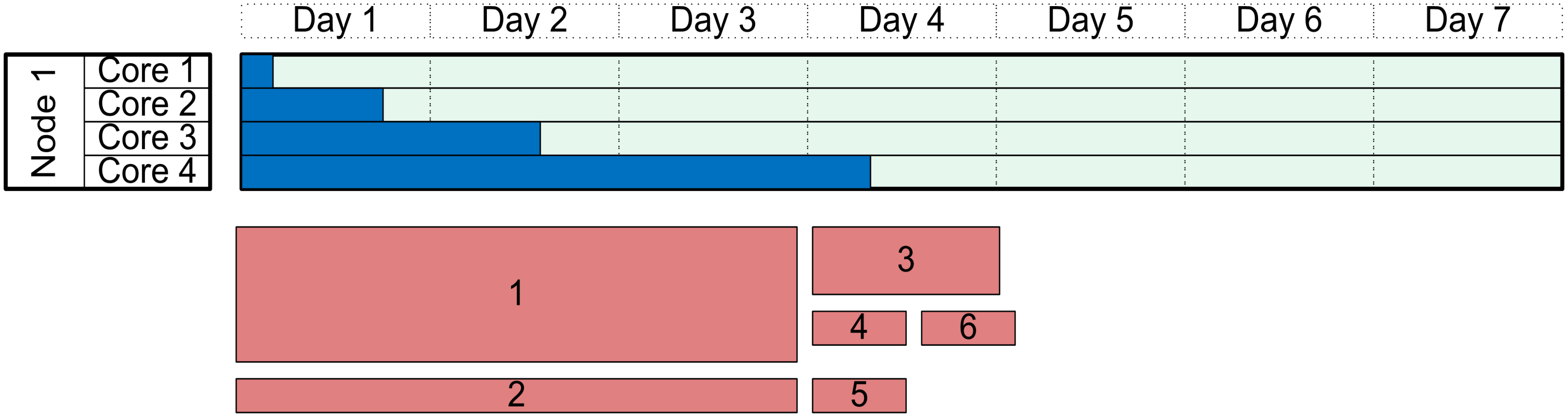
Core 3

Core 4

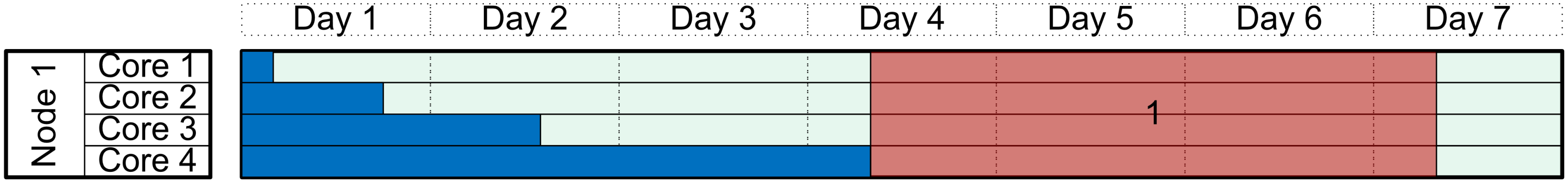
Running jobs



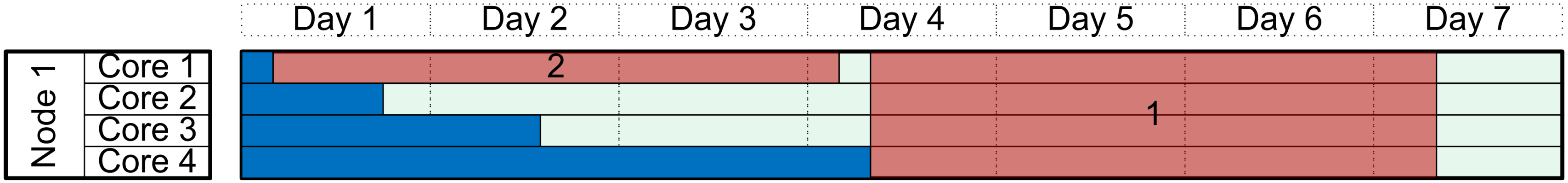
Scheduling jobs in order of priority



Scheduling jobs in order of priority



Scheduling jobs in order of priority



Day 1 Day 2 Day 3 Day 4 Day 5 Day 6 Day 7

Node 1

Core 1

Core 2

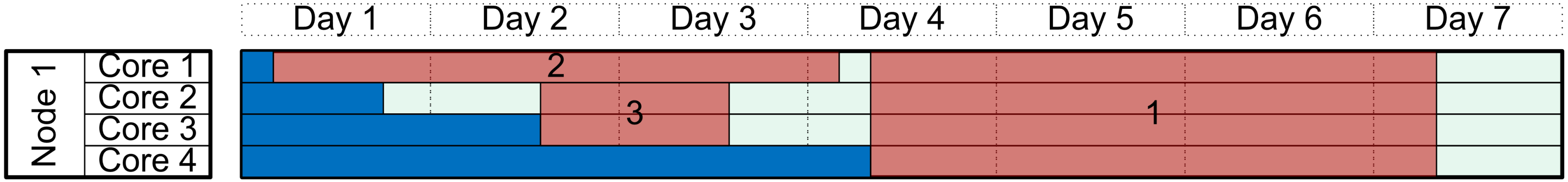
Core 3

Core 4

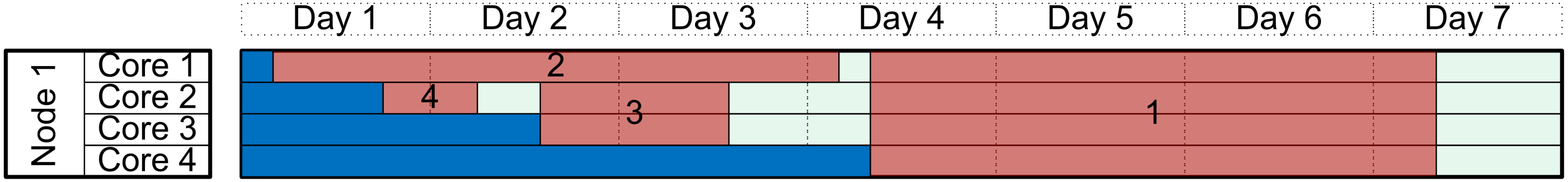
2

1

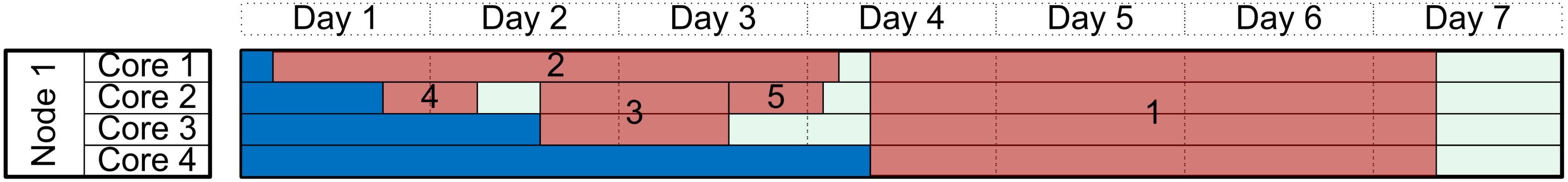
Scheduling jobs in order of priority



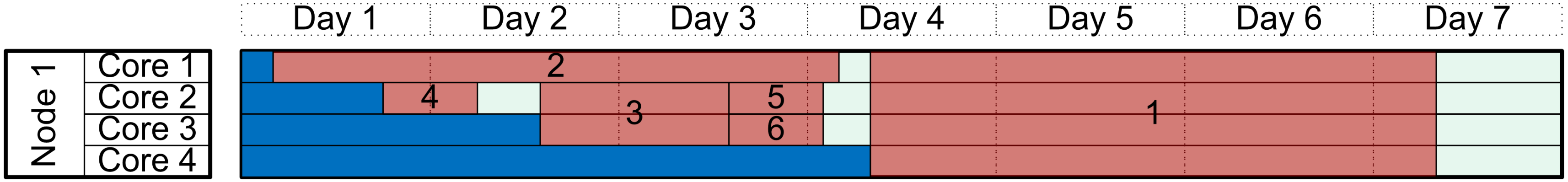
Scheduling jobs in order of priority



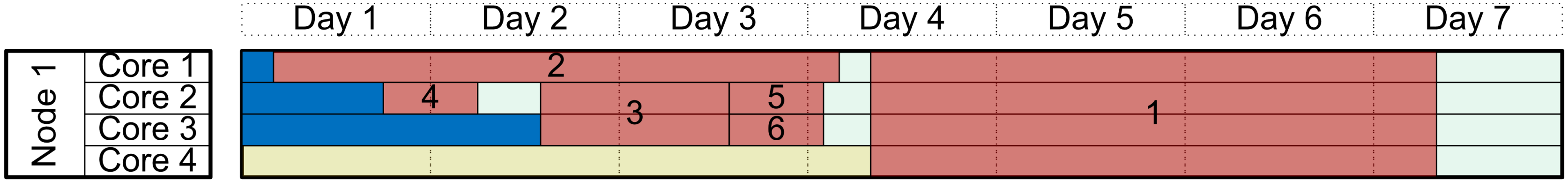
Scheduling jobs in order of priority



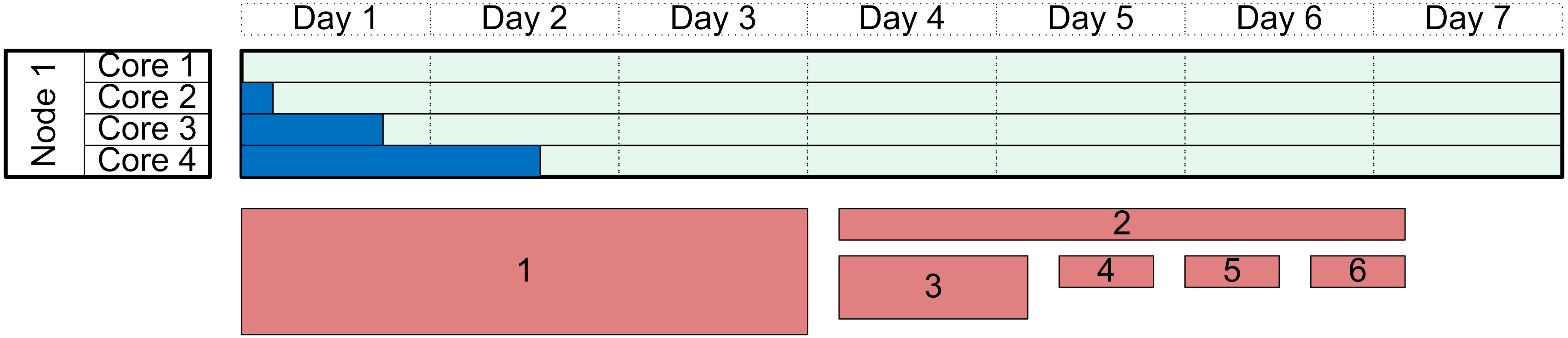
Scheduling jobs in order of priority



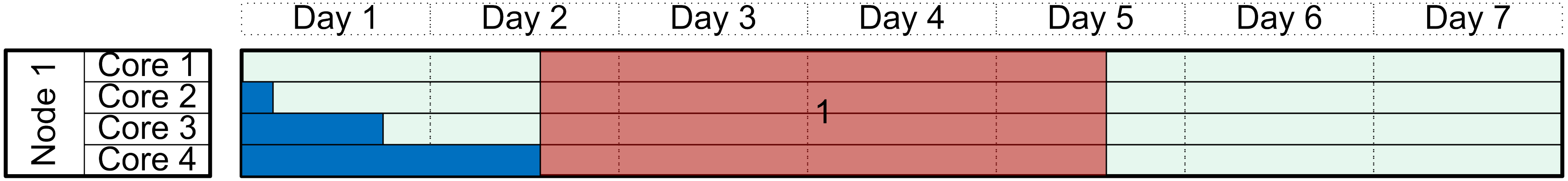
A Job finishes early



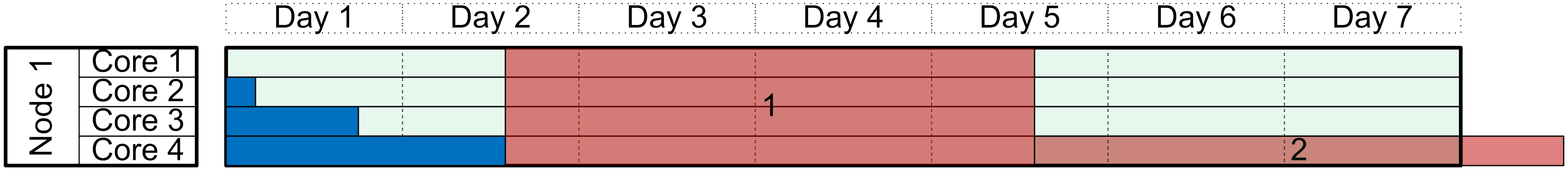
Jobs are rescheduled



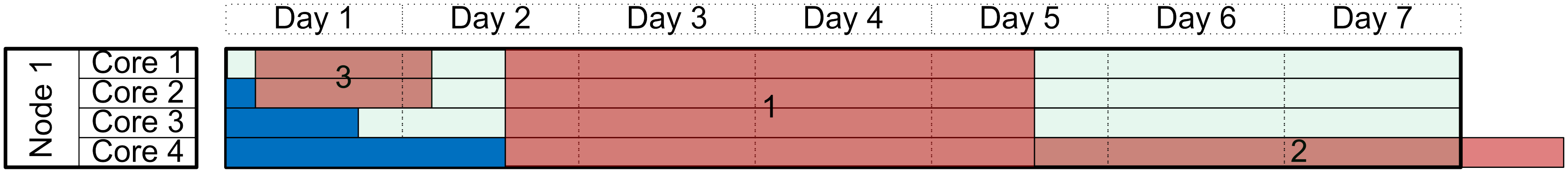
Jobs are rescheduled



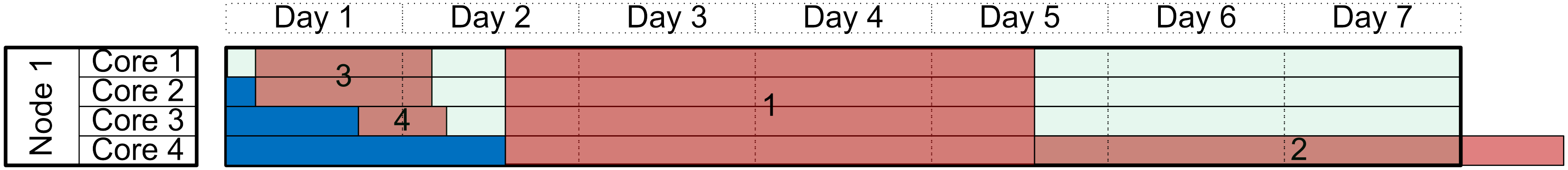
Jobs are rescheduled



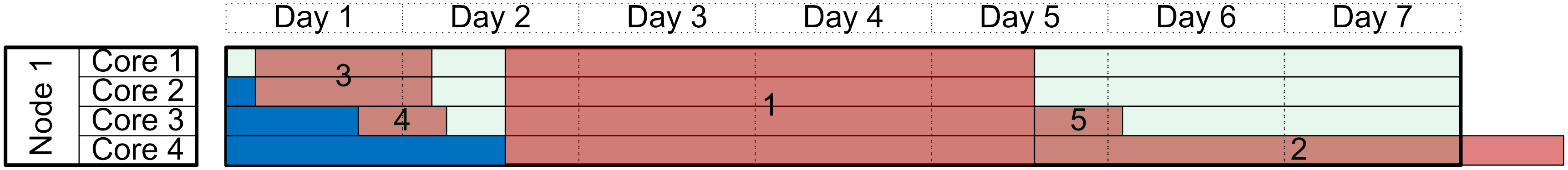
Jobs are rescheduled



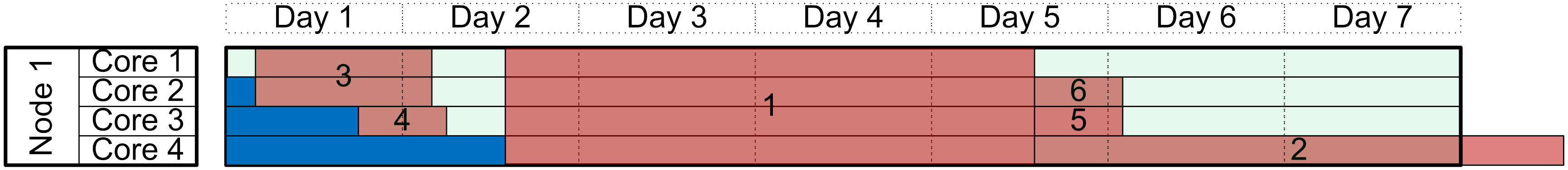
Jobs are rescheduled



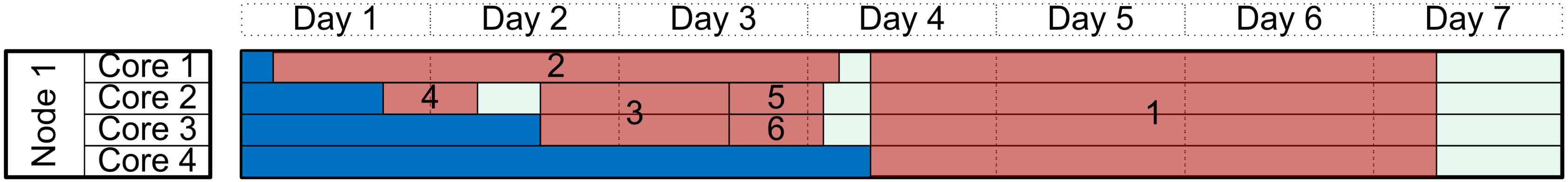
Jobs are rescheduled



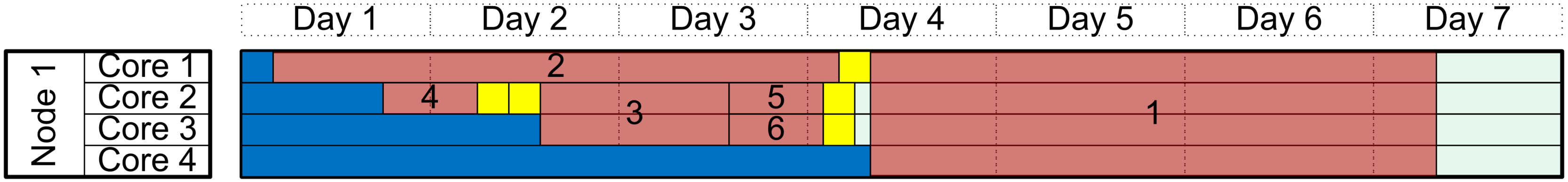
Jobs are rescheduled



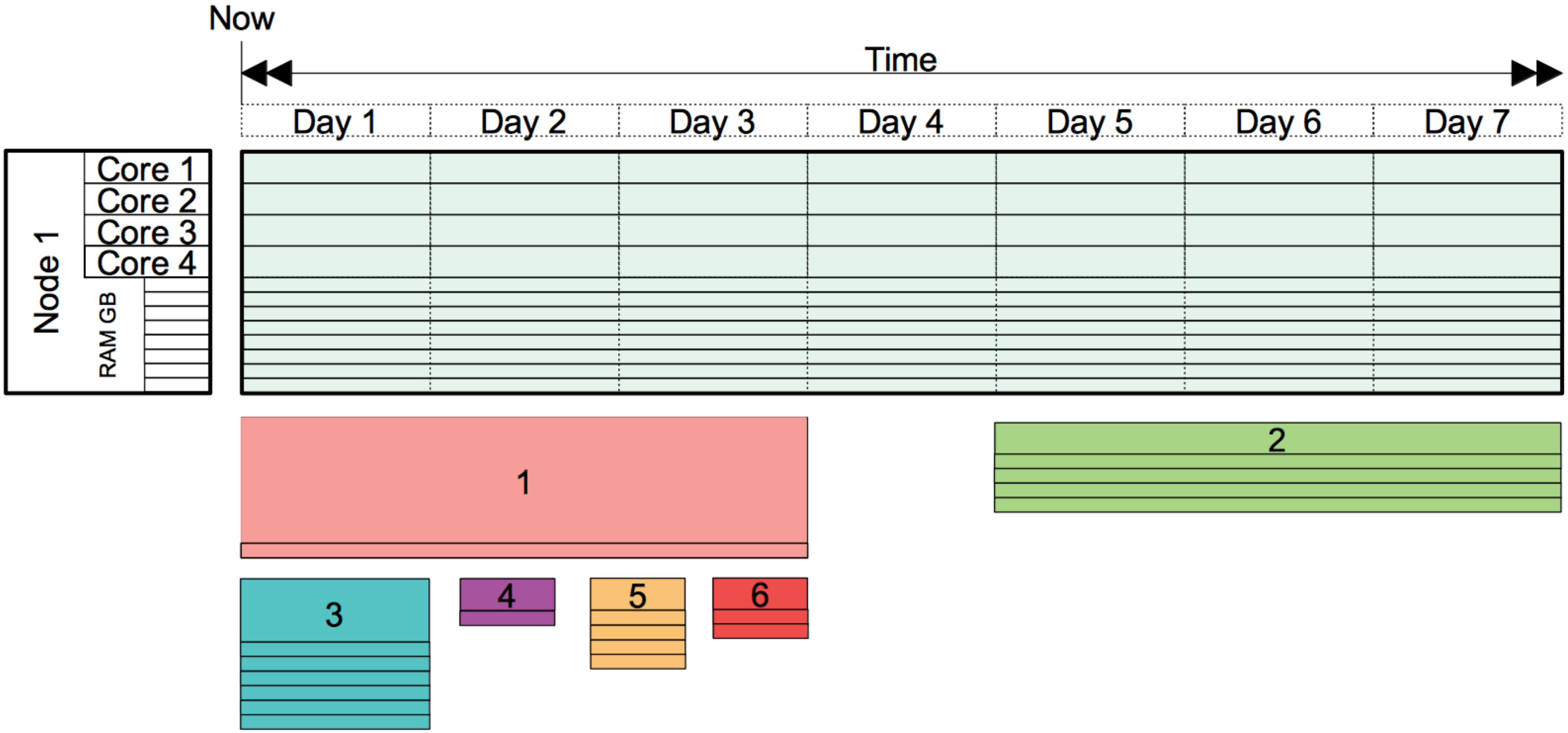
Single node cluster



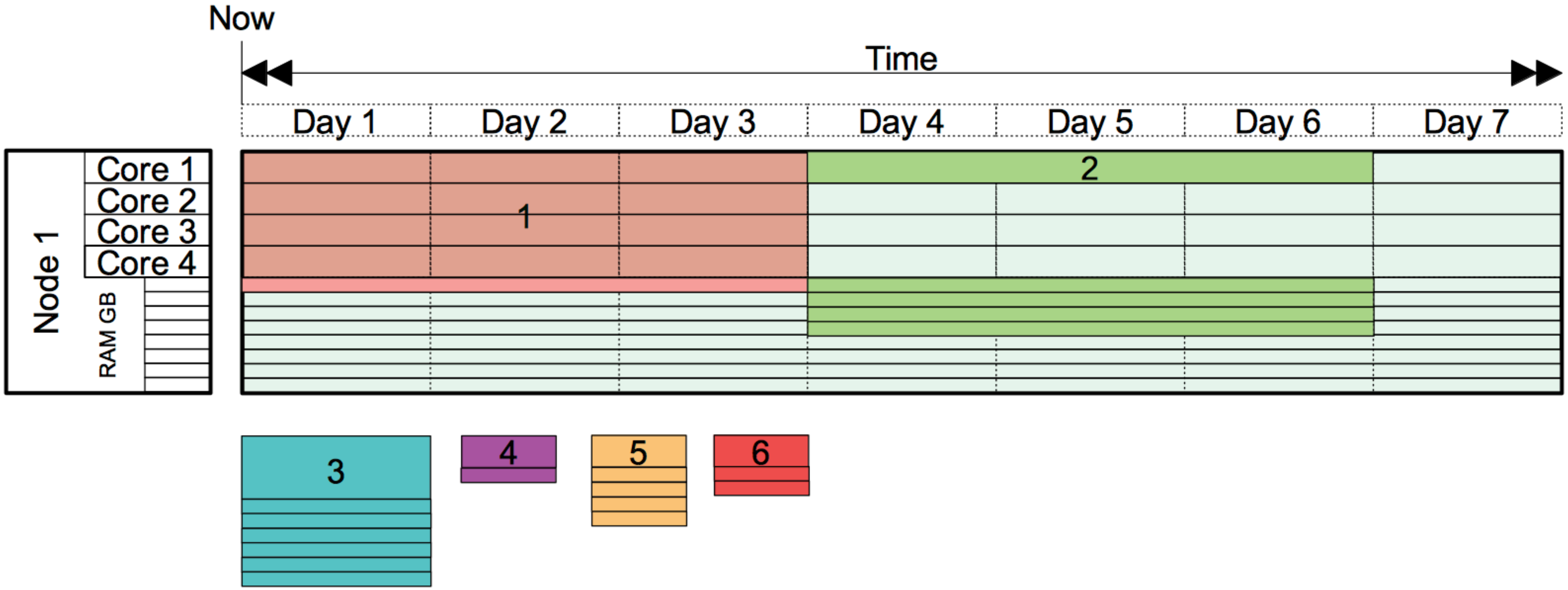
Short serial jobs and Backfill



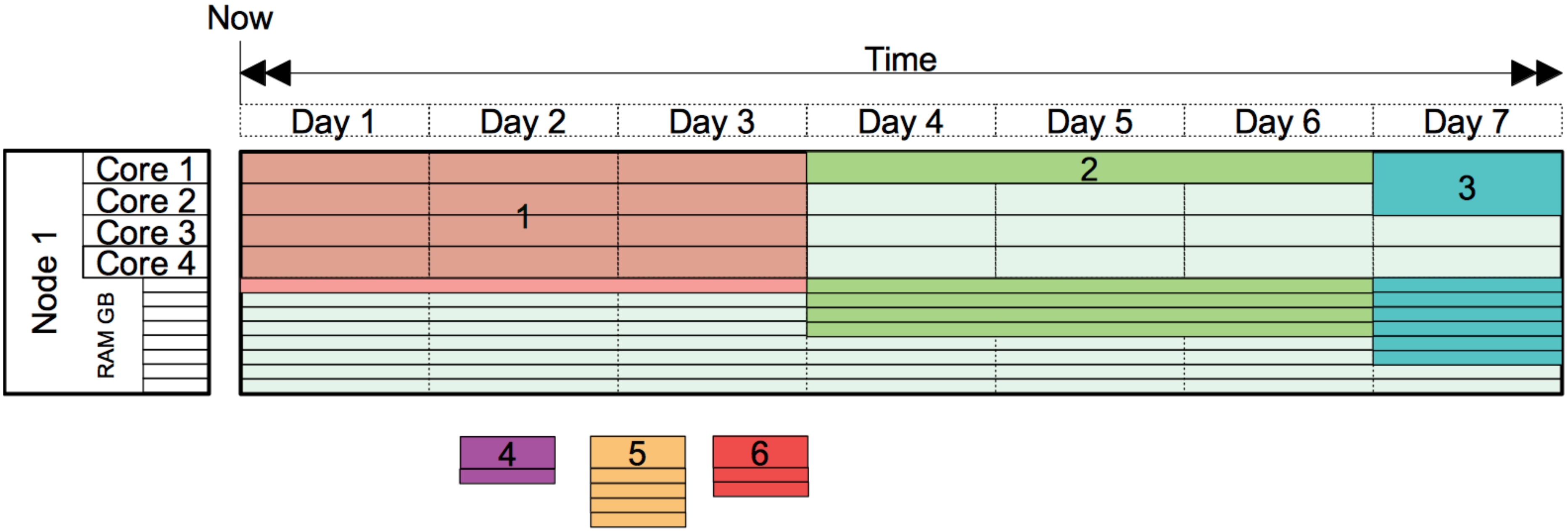
Scheduling Cores and Memory



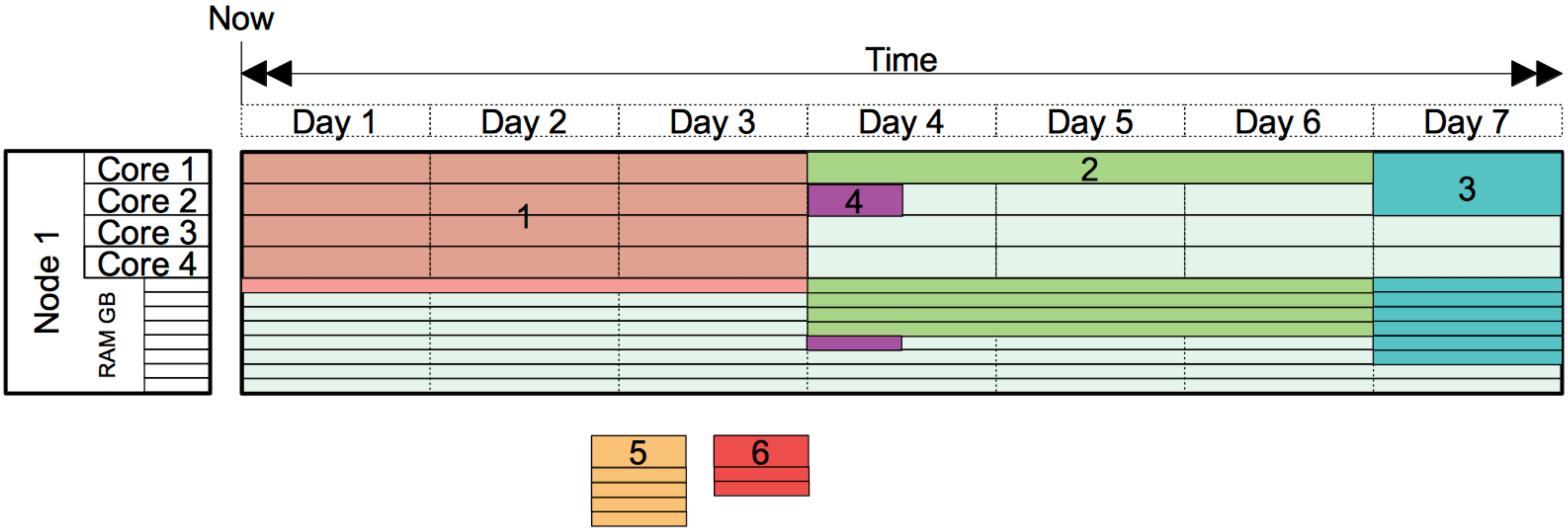
Scheduling Cores and Memory



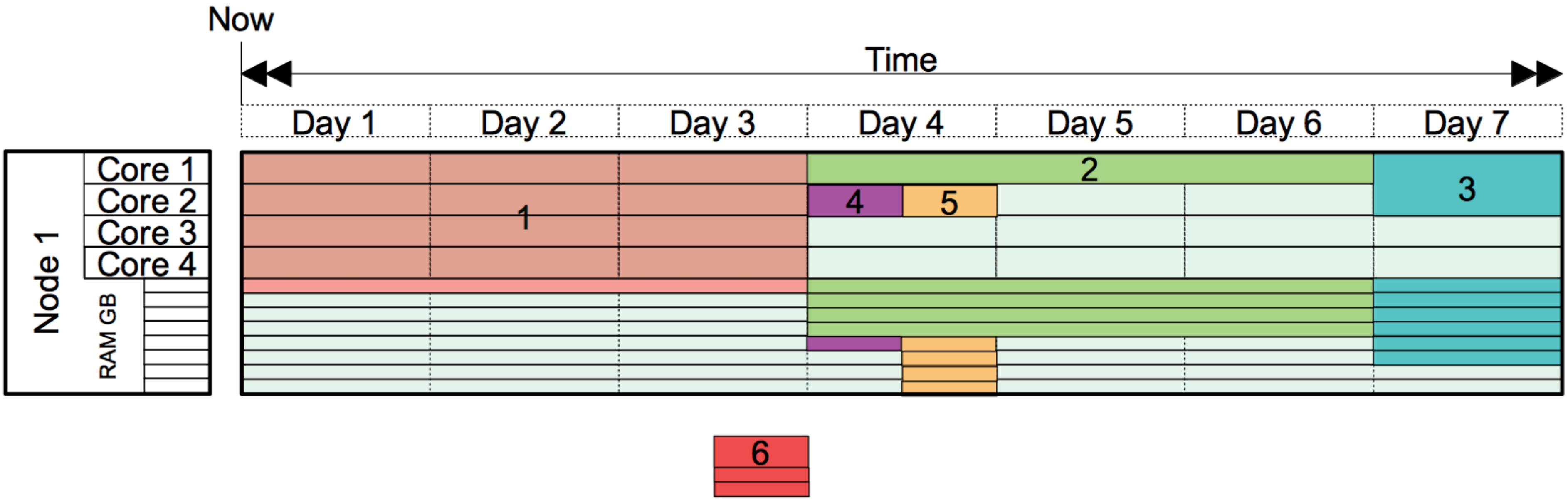
Scheduling Cores and Memory



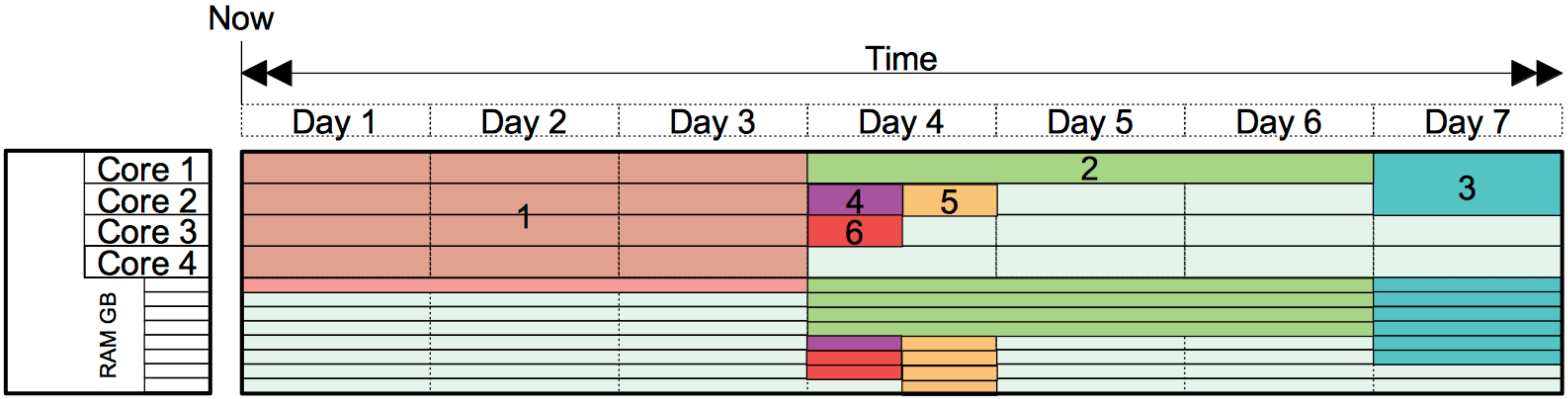
Scheduling Cores and Memory



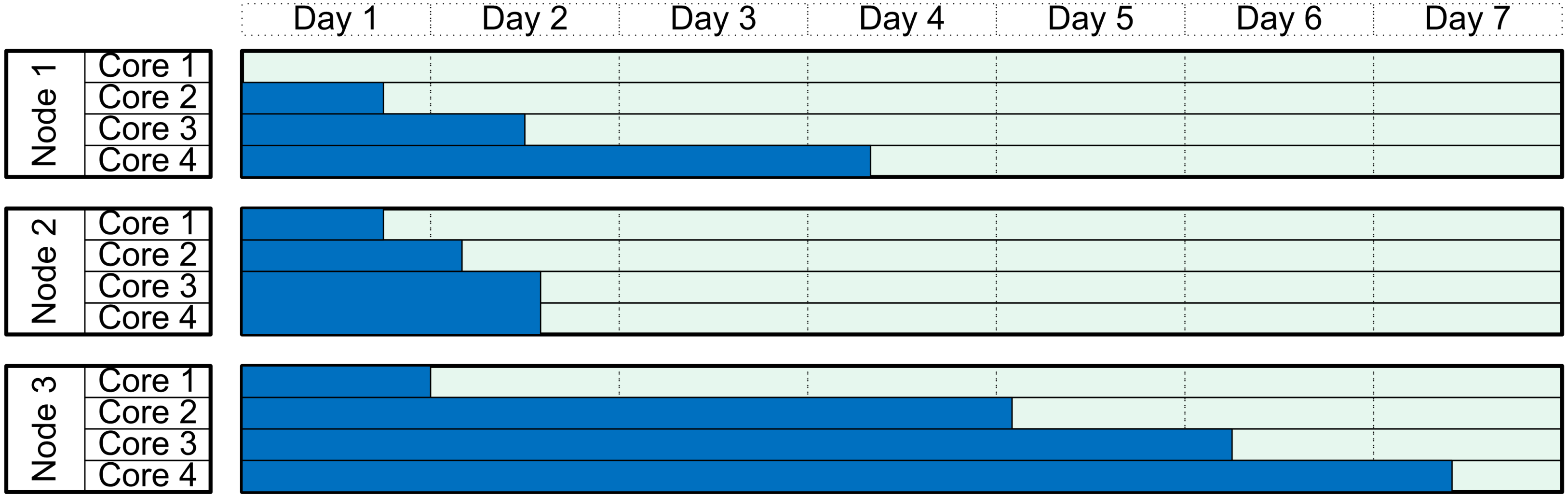
Scheduling Cores and Memory



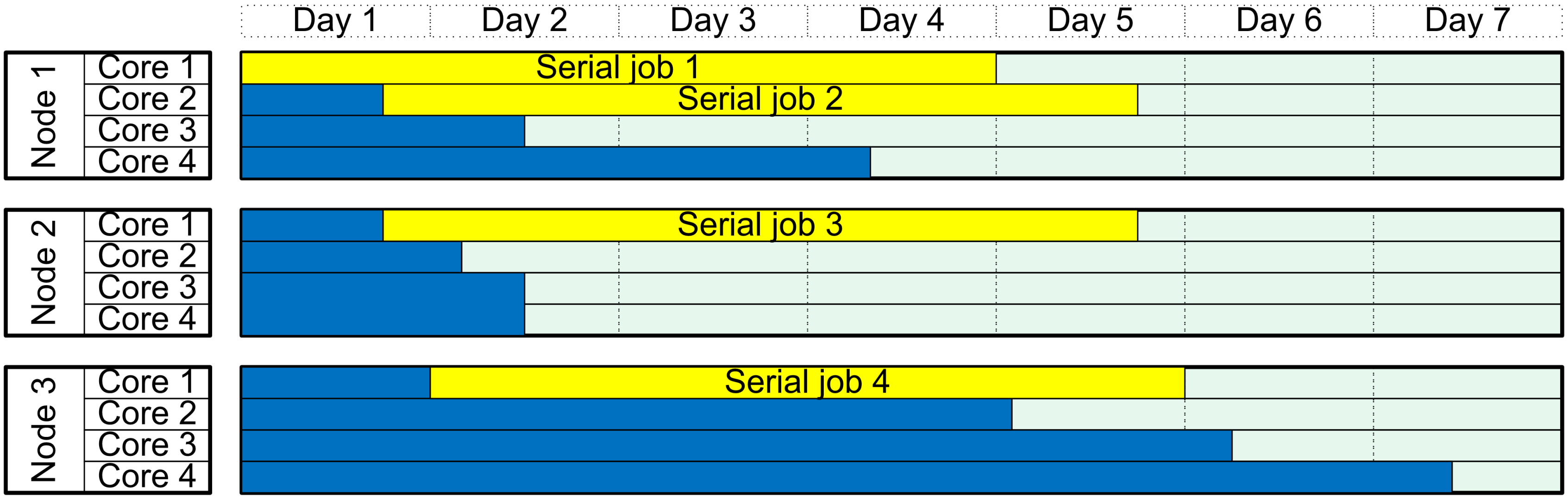
Scheduling Cores and Memory



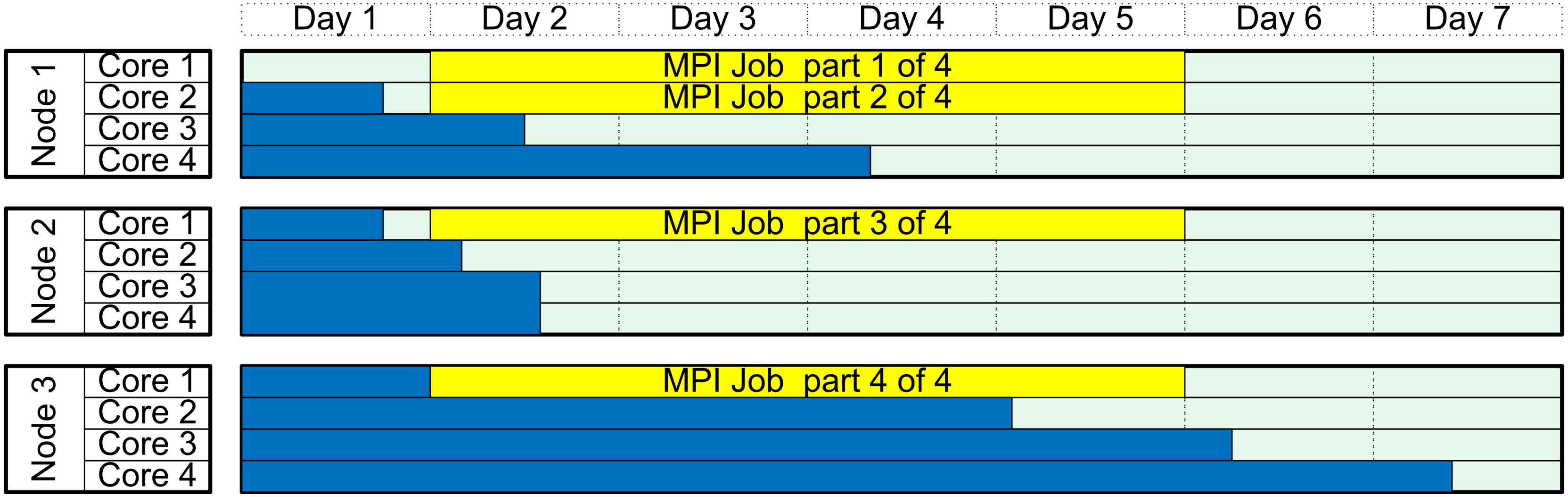
Visualizing Multinode cluster



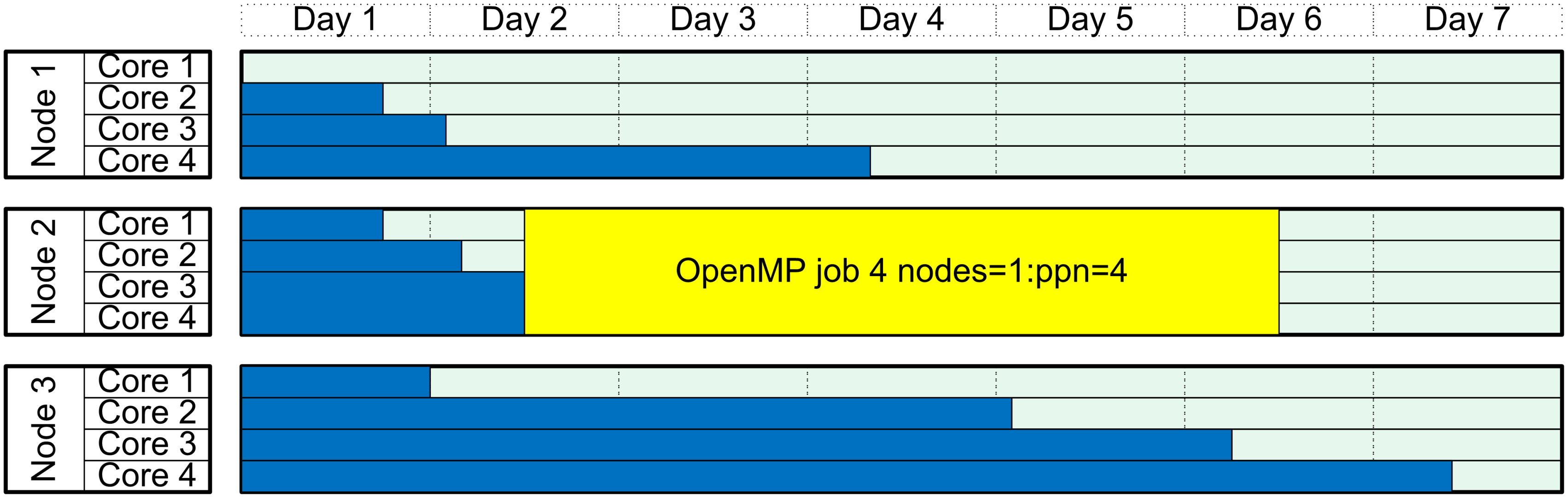
Many Serial Jobs



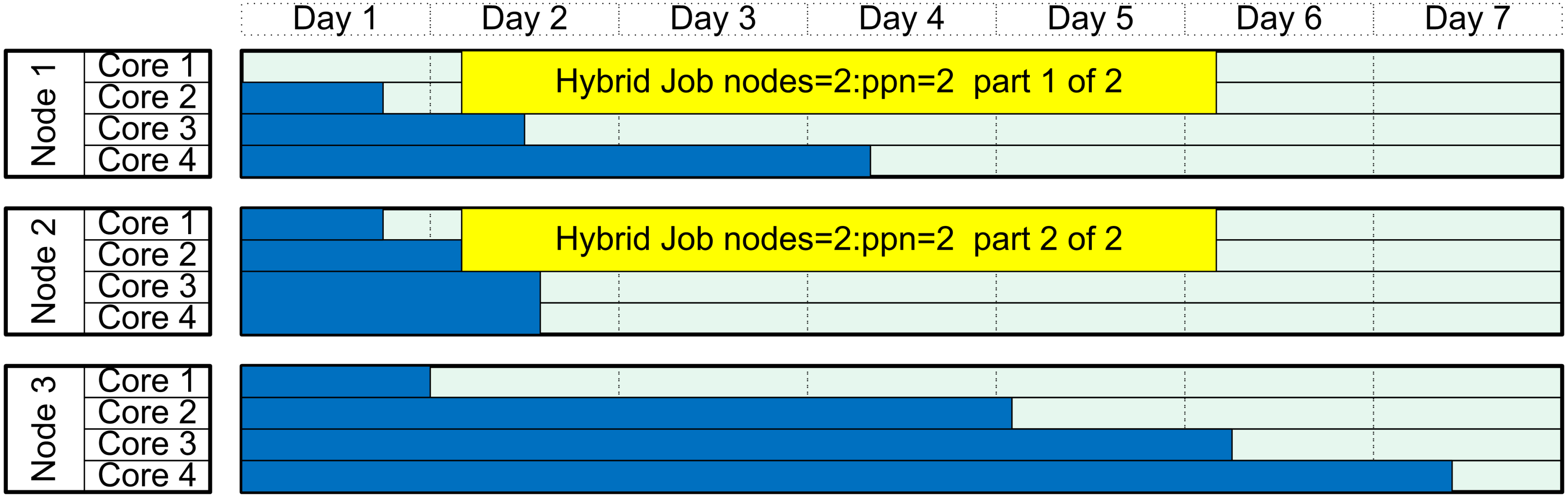
MPI job


Day 1
Day 2
Day 3
Day 4
Day 5
Day 6
Day 7
Node 1
Core 1
Core 2
Core 3
Core 4
MPI Job part 1 of 4
MPI Job part 2 of 4
Node 2
Core 1
Core 2
Core 3
Core 4
MPI Job part 3 of 4
Node 3
Core 1
Core 2
Core 3
Core 4
MPI Job part 4 of 4

Single node muti-core job (OpenMP, Gaussian, Threads)



Hybrid Job


Day 1
Day 2
Day 3
Day 4
Day 5
Day 6
Day 7
Node 1
Core 1
Core 2
Core 3
Core 4
Hybrid Job nodes=2:ppn=2 part 1 of 2
Node 2
Core 1
Core 2
Core 3
Core 4
Hybrid Job nodes=2:ppn=2 part 2 of 2
Node 3
Core 1
Core 2
Core 3
Core 4

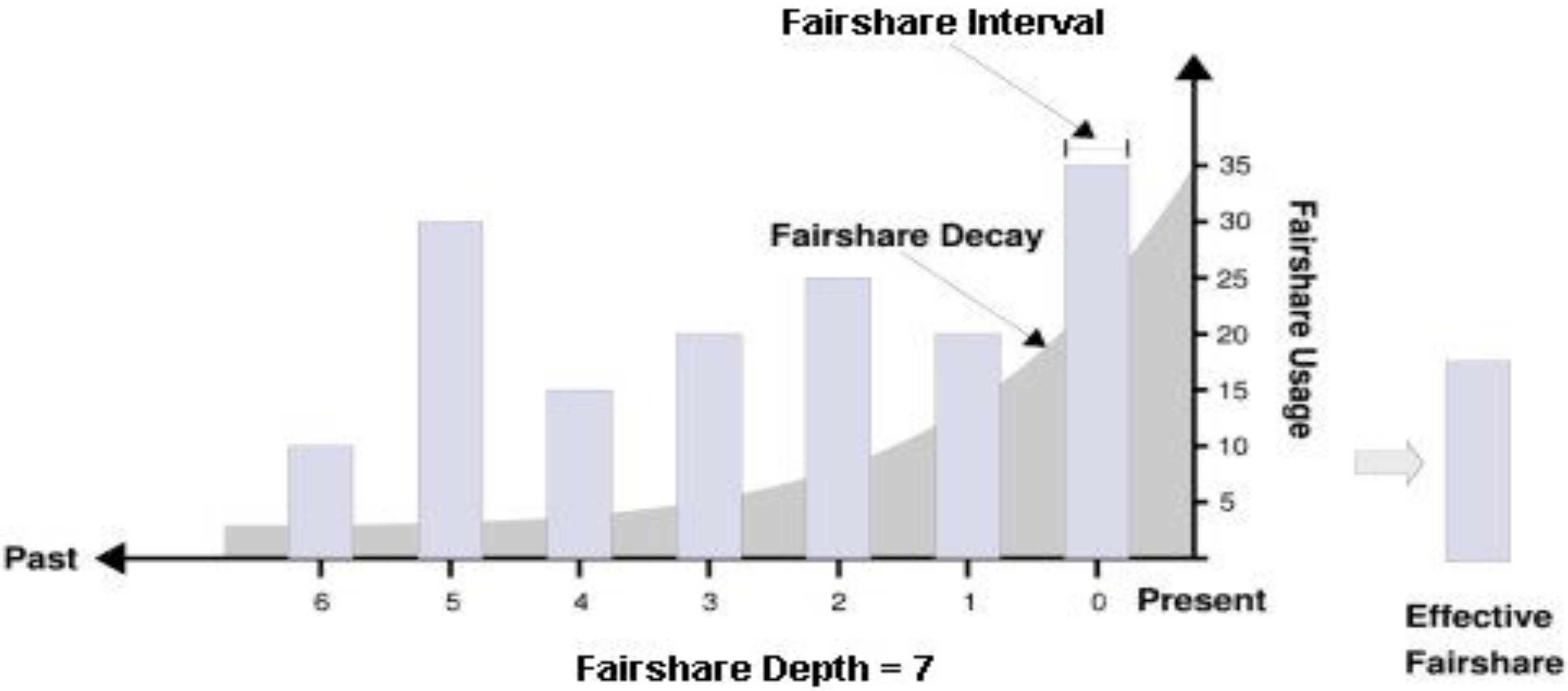
Maximum job walltime partition limit

- A high maximum walltime is not necessary a good thing, clusters that allow high walltime jobs take longer for jobs to start to run, and are less “fair”.
- There are advantages to running shorter jobs, such as how quickly your job can be started.
- The longer and larger a job is the greater the chances of experiencing hardware failure, minimize this through check pointing.
- Part of the resources of a cluster is dedicated for shorter jobs.

Partition name	Maximum walltime
*_b1	3 hours
*_b2	12 hours
*_b3	1 day
*_b4	3 days
*_b4	7 days
*_b6	28 days

- Part of CC clusters are dedicated to whole node parallel jobs, other jobs with a short walltime of under 12 hours can run in this part at a reduced priority compared to whole node parallel jobs.

Fairshare



- Fair share usage is weighted by when the usage occurred recent usage is more important then usage at the end of the period

Group’s Status: “sshare -l”

```
[kamil@cedar5 workshop_test]$ sshare  -l | egrep "(--|Account|^root|no_rac_|ras_b|cc-debug|kamil_)"
             Account       User  RawShares  NormShares    RawUsage   NormUsage  EffectvUsage  FairShare LevelFS
-------------------- ---------- ---------- ----------- ----------- ----------- ------------- ---------- ----------
root                                          0.000000 639083114320110                  1.000000
 no_rac_cpu                           1320    0.043194 404703982221822    0.633257      0.633257              0.068209
  ras_basic_cpu                       1320    0.999243 404703982221822    0.633257      1.000000              0.999243
   cc-debug_cpu                          1    0.000236  1273287234    0.000002      0.000003             75.104409
    cc-debug_cpu          kamil          1    0.004386           0    0.000000      0.000000   0.026537        inf
   def-kamil_cpu                         1    0.000236           0    0.000000      0.000000                   inf
    def-kamil_cpu         kamil          1    1.000000           0    0.000000      0.000000   0.486678        inf
 no_rac_gpu                             65    0.002127 6883285083841    0.010771      0.010771              0.197479
  ras_basic_gpu                         65    0.984848 6883285083841    0.010771      1.000000              0.984848
   cc-debug_gpu                          1    0.000236       12668    0.000000      0.000000            128389.386733
    cc-debug_gpu          kamil          1    0.004386           0    0.000000      0.000000   0.508693        inf
   def-kamil_gpu                         1    0.000236           0    0.000000      0.000000                   inf
    def-kamil_gpu         kamil          1    1.000000           0    0.000000      0.000000   0.973463        inf
```

Jobs by partition

squeue -p <partitionname>

```
JOBID PARTITION     NAME     USER ST        TIME  NODES NODELIST(R
535639 cpubase_b AE17631.    kamil PD       0:00      1 (Resource
591830 cpubase_b    bz.sh   erming PD       0:00      1 (Resource
615762 cpubase_b AE21380.    kamil PD       0:00      1 (Resource
401219 cpubase_b CTD095.s     john PD       0:00      1 (Resource
491576 cpubase_b gen3x1s8     judy  R 2-08:04:59      1 cdr747
535638 cpubase_b AE17594.    kamil  R 1-11:46:03      1 cdr101
491574 cpubase_b gen3x1s6    masao  R 4-20:06:44      1 cdr79
491575 cpubase_b gen3x1s7    masao  R 4-20:06:44      1 cdr85
```

Priority
sprio

```
 JOBID   PRIORITY        AGE  FAIRSHARE  PARTITION                 TRES
130976       7088       2500          0        625    cpu=2526,mem=1437
167003       6150       2500          0       1250     cpu=2008,mem=392
195802    4996086       2500    4991771        833 cpu=469,mem=45,gres/
195809    4996086       2500    4991771        833 cpu=469,mem=45,gres/
195810    4996086       2500    4991771        833 cpu=469,mem=45,gres/
205281       8206       2500          0        625 cpu=1875,mem=1800,gr
205290       6408       2500          0        625 cpu=1875,mem=2,gres/
544814      23534       1741      21571        208         cpu=13,mem=2
544815      23534       1741      21571        208         cpu=13,mem=2
617580      24194        373      22768       1042         cpu=10,mem=2
617581      24194        373      22768       1042         cpu=10,mem=2
```