HPC Carpentry Essentials

Distributing resources without scheduling exercise.

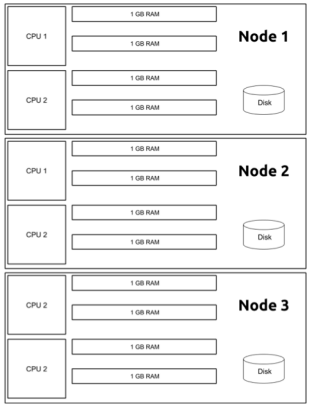
In this example we will be dividing cluster for the entire year, not via batch scheduler

There is more than one right answer.

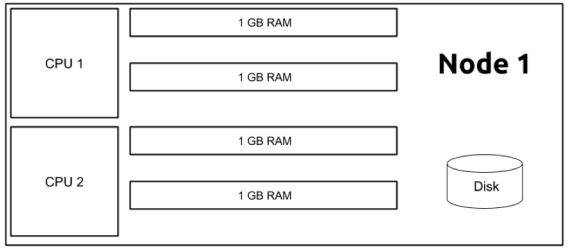
You will be give a sheet of paper representing a cluster and a ziplock bag with colored squares and rectangles.

The color of the shape identifies to whom it belongs.

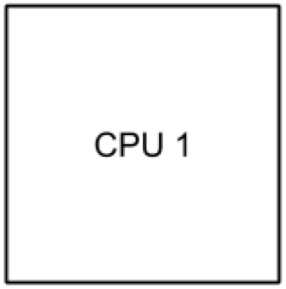
Diagram of Materials



Cluster



Node



Core

RAM

Distributing Resources

You are appointed to run a HPC facility in a fictional nation (CW).

You have a number of limited resources 3 nodes with 2 cores each.

You also have requests from number of users with compute needs.

Distributing Resources

Users need to guarantee that they will actually get any allocation that they are given.

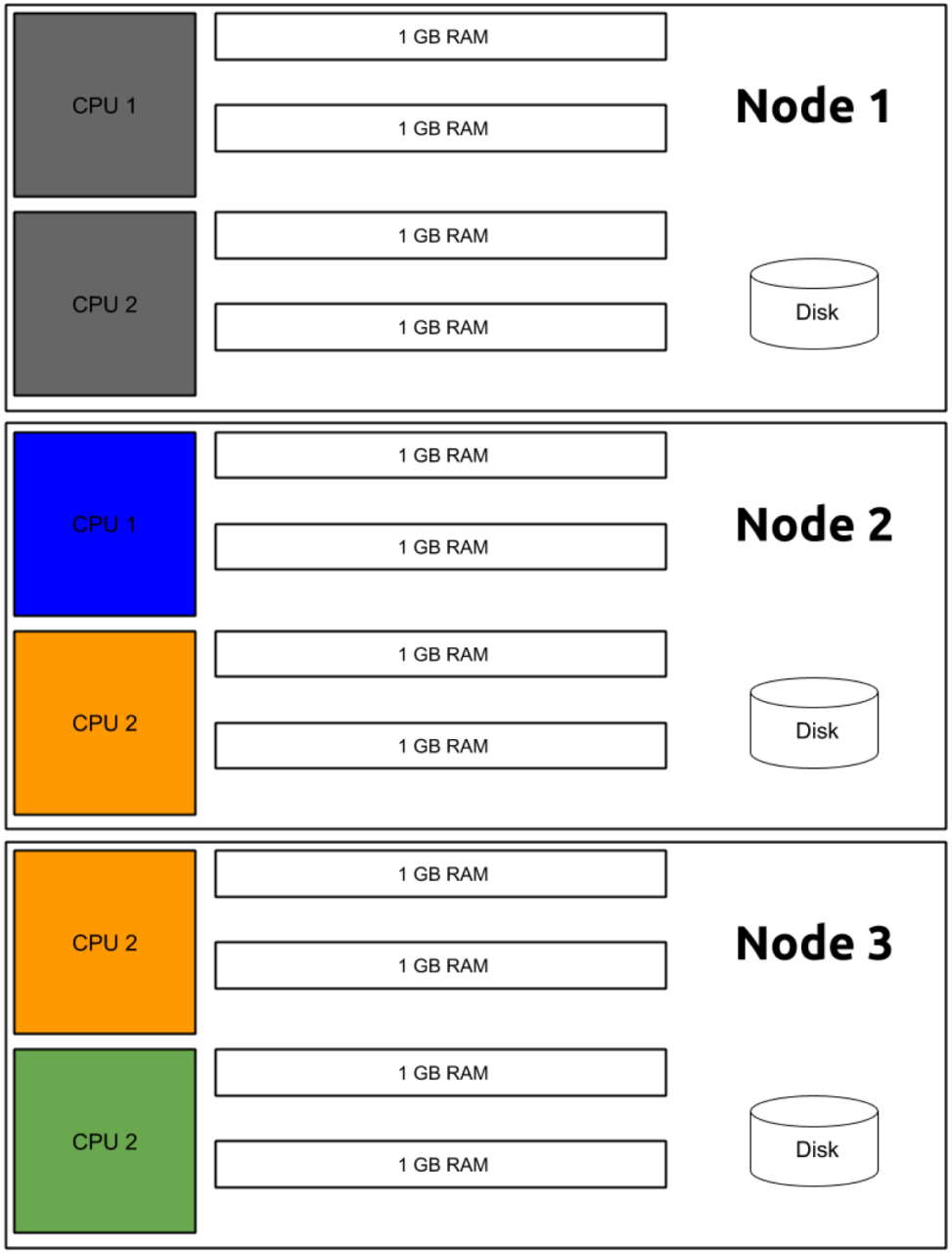
Discus among group: Users need to request resources what questions do you ask?

Please allocate the requested resources (cores) in this instance by putting the shapes onto the sheet of paper representing the cluster.

Please use the representative color for each of the users jobs.

Look at your colleagues allocations and discuss the choices made and why.

Needs				
Name	Color	Priority	Jobs	Cores per Job
T'Challa	**Black**	Highest	2	1
Steve	**Blue**	High	1	1
Tony	**Orange**	Med+	2	1
Bruce	**Green**	Med	1	1
Stephen	**Purple**	Low	1	1



Distributing resources

One possible solution

Distributing Resources part 2

You are appointed to run a HPC facility in a fictional nation (CW).

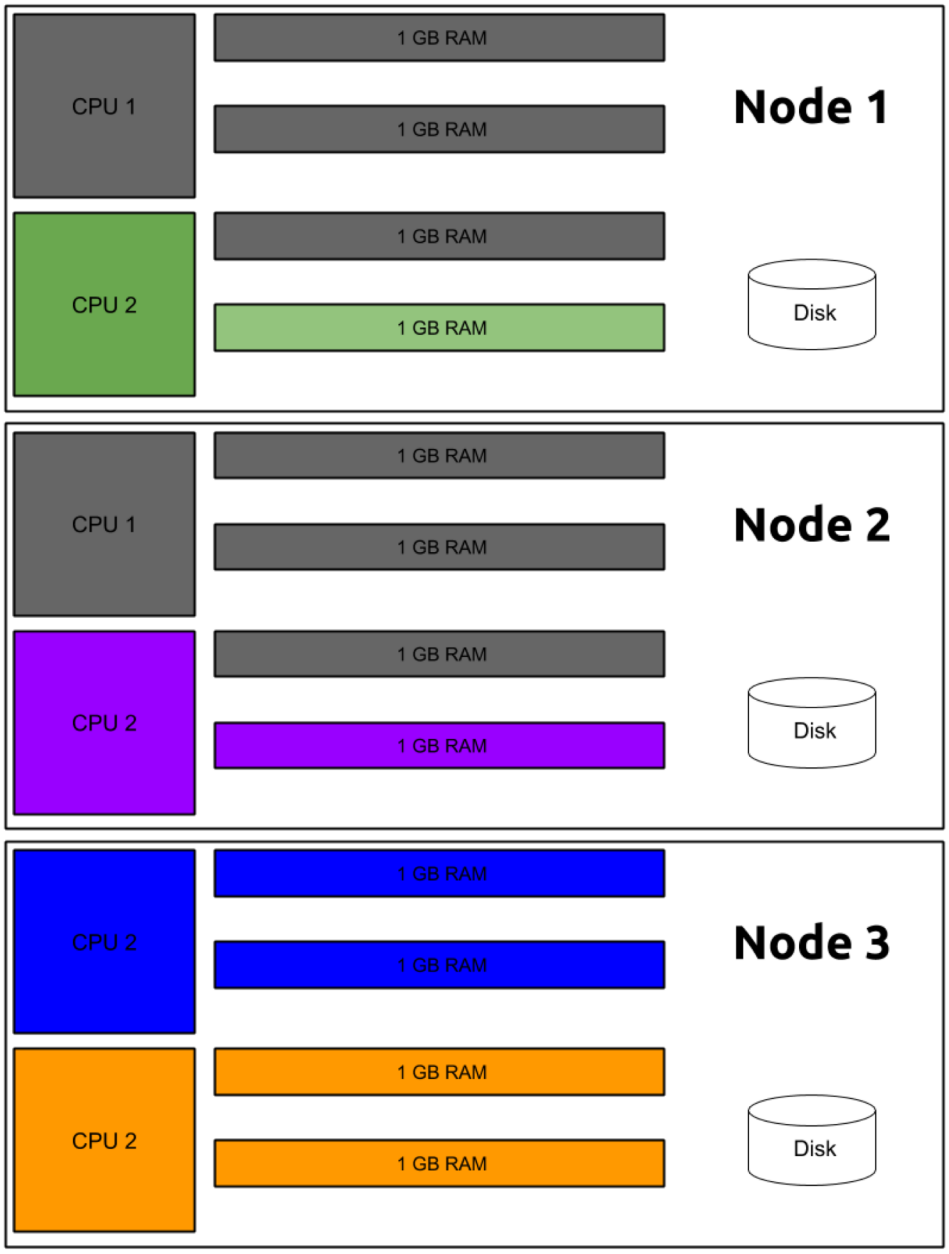
You have a number of limited resources 3 nodes with 2 cores and **4 GB RAM each**.

You also have requests from number of users with compute needs.

Users need to guarantee that they will actually get any allocation that they are given.

Users need to request resources what questions do you ask?

Needs					
Name	Colour	Priority	Jobs	Cores per Job	**Mem/core GB**
T'Challa	**Black**	Highest	2	1	3
Steve	**Blue**	High	1	1	2
Tony	**Orange**	Med+	2	1	2
Bruce	**Green**	Med	1	1	1
Stephen	**Purple**	Low	1	1	1



Distributing resources part 2

One possible solution

How do we (CW) take requests for resources?

In a Group discuss along yourself:

- You have a large number of resources and a large number of request.
- How do you take requests for resources,
  - Remember you have to guarantee that you can have enough resources to give to the people that you have granted allocation.
  - What questions do you ask? What process do you use to make sure you allocate as many high priority users with resources as possible?

Distributing Resources and priority

- There are multiple resources cores, memory, GPUs etc…
- In order to be used most resources must be used in conjunction with other ie: A computation that requires a GPU also requires use of some CPU cores and system memory.
- As a consequence priority must be one number. A computation cannot have a RAM priority and a different Core priority, as the priority determines if the computation runs. The use of cores and memory both need to happen and at the same time for the computation to take place.

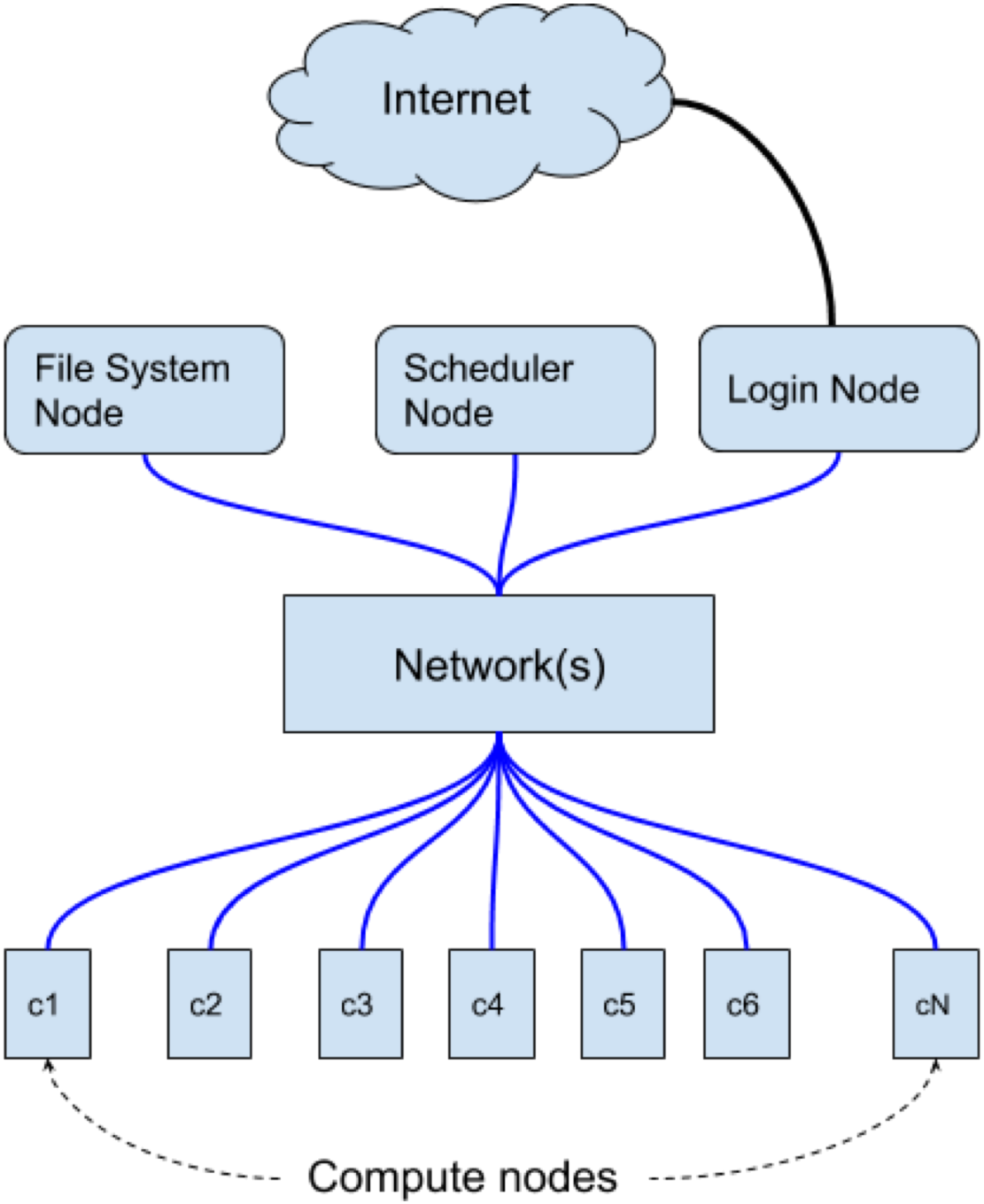
Core Equivalent

- The Scheduling system has a priority system based upon how many resources used by each group in the past. We need a single number to represent a groups usage as a share of cluster resources.
- Each computation uses a number of resources. We count the usage as the portion of the maximum share of resources of a typical node.
- Ie if a typical cluster node has 32 cores and 128 GB of RAM. Job A uses 1 core and 96 GB RAM (3/4 of RAM on the node) it can be said to use ¾ of a Node, or expressed as the equivalent of ¾ of the cores on one Node. So we can say that Job A uses 32* ¾ = 24 Core Equivalent. We would use this number In determining usage, priority, and allocations.

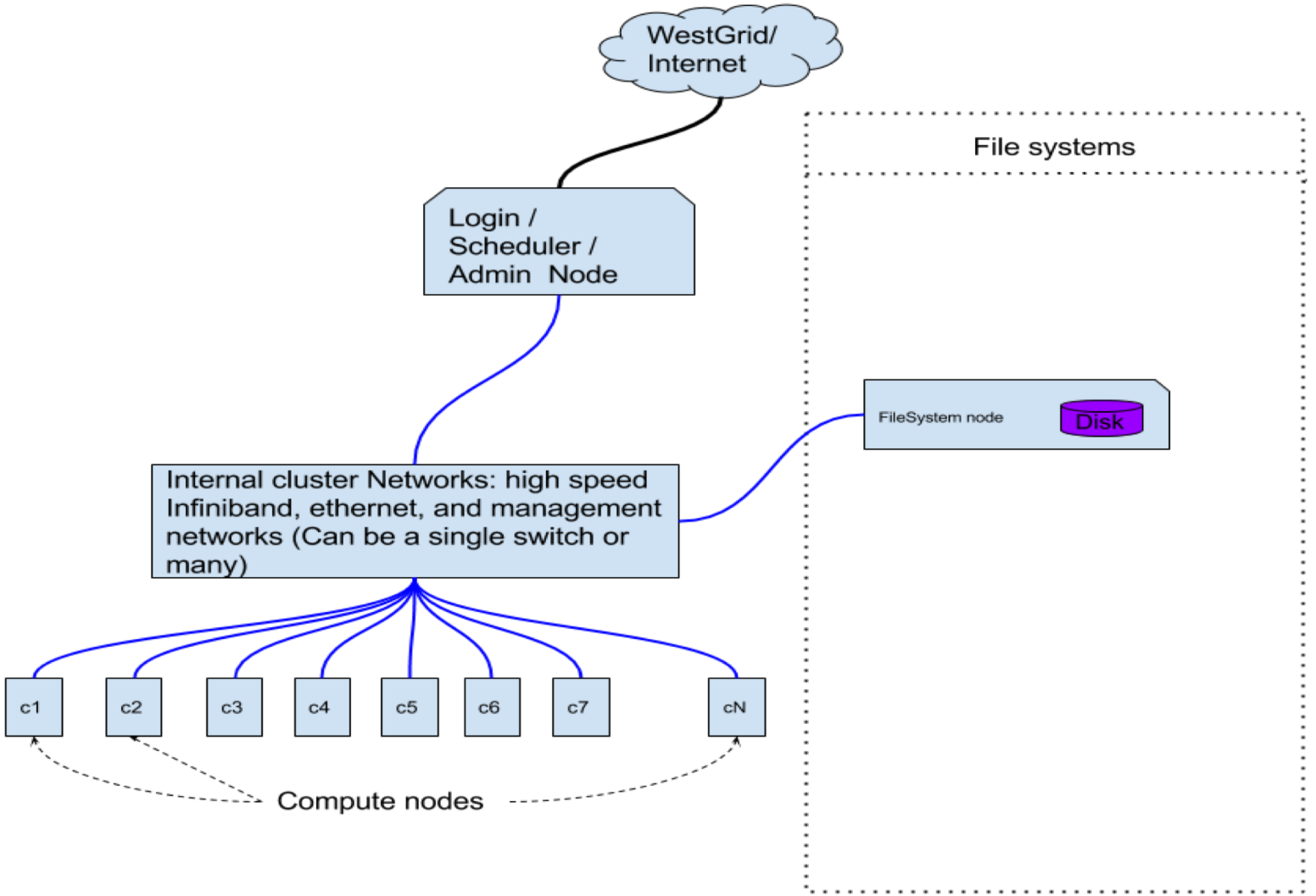
Batch Scheduling

- Is not used when you need a service for example a webserver that runs all the time.
- Is preferred when you have one or more jobs (simulations) that need to be run and you wish to get the results back sometime in the future.
- Your job automatically started by the scheduler when enough resources are available, and you get results back, you may be notified when your job starts and finishes.

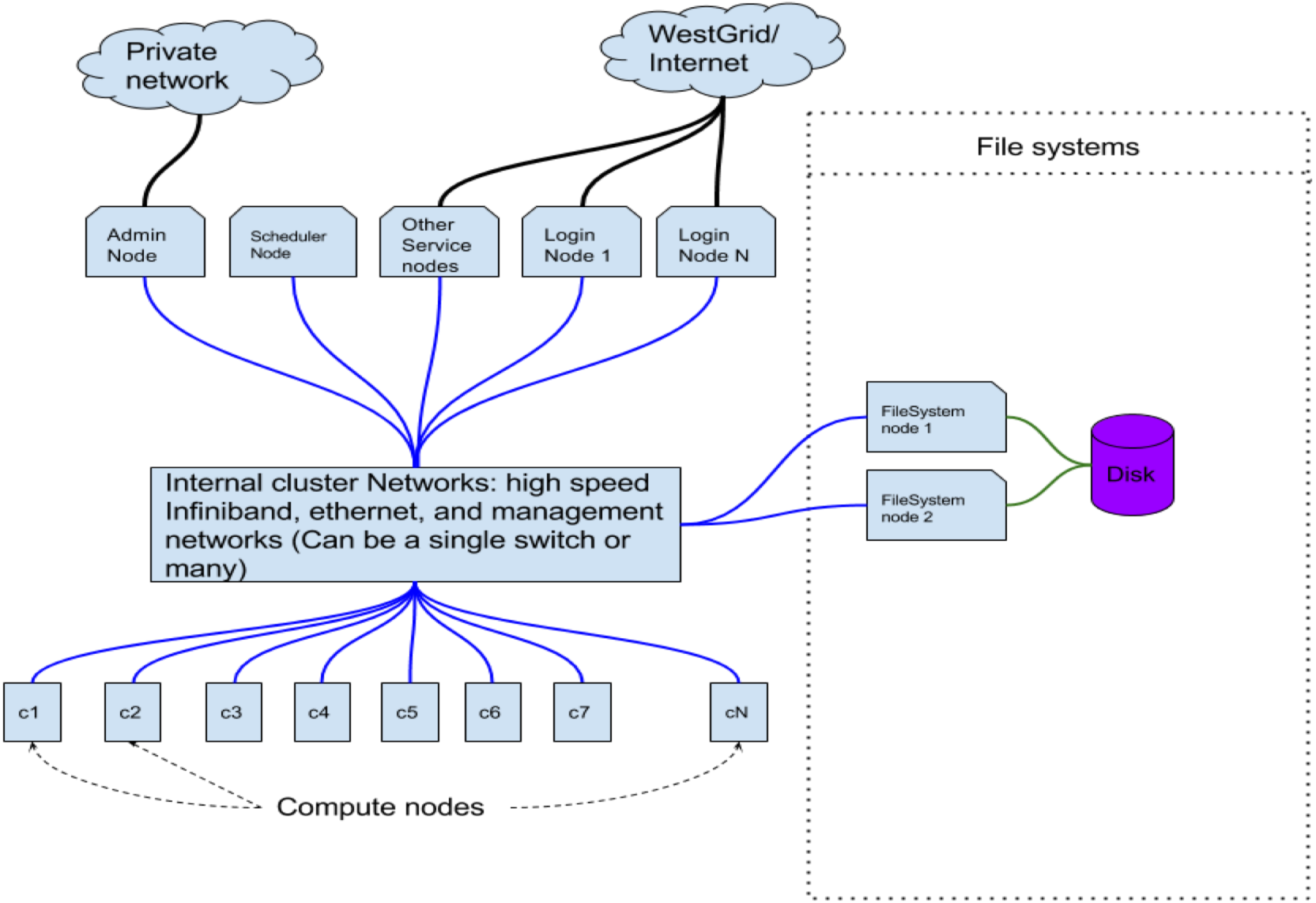
Cluster Diagram



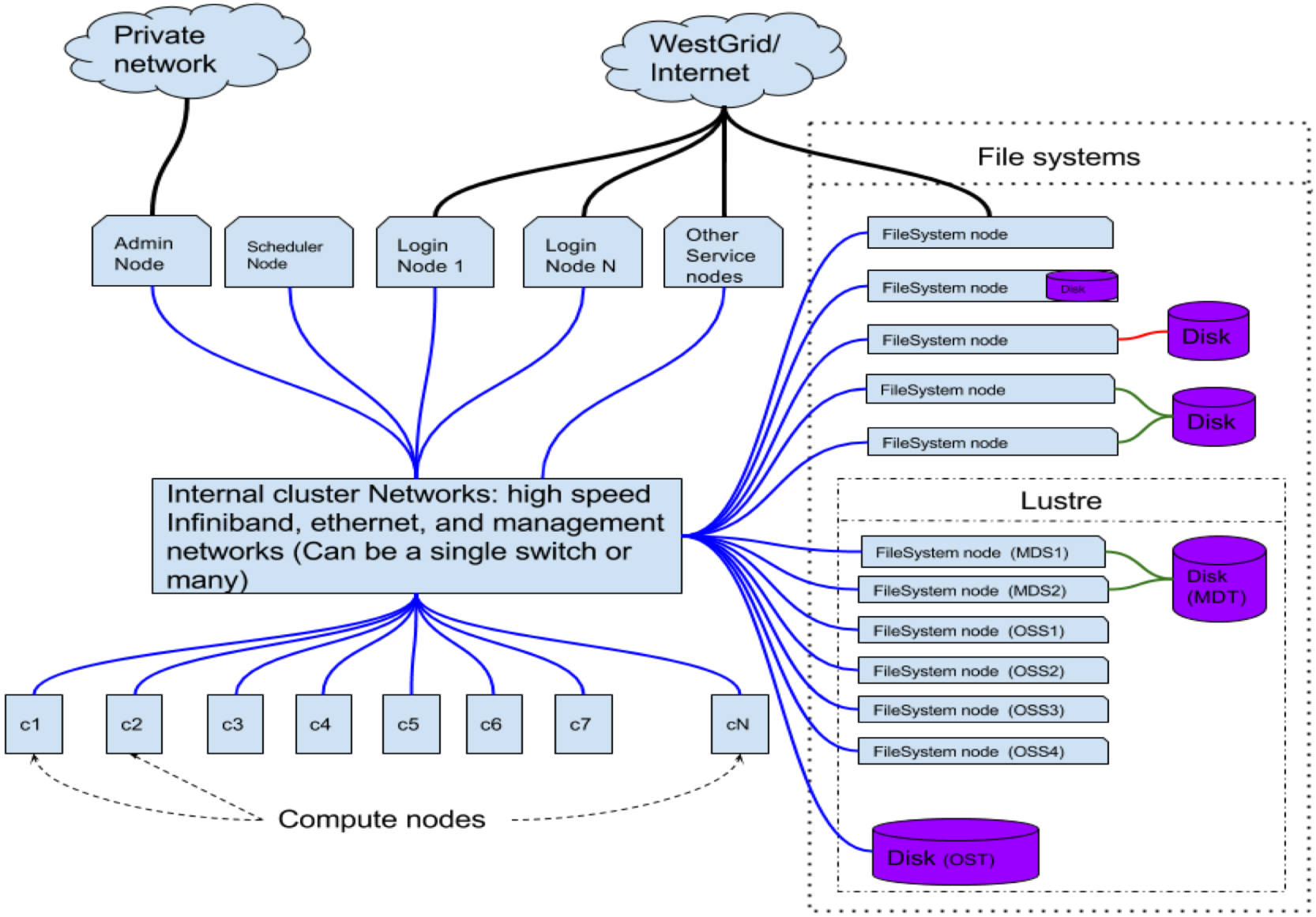
Typical small HPC Cluster



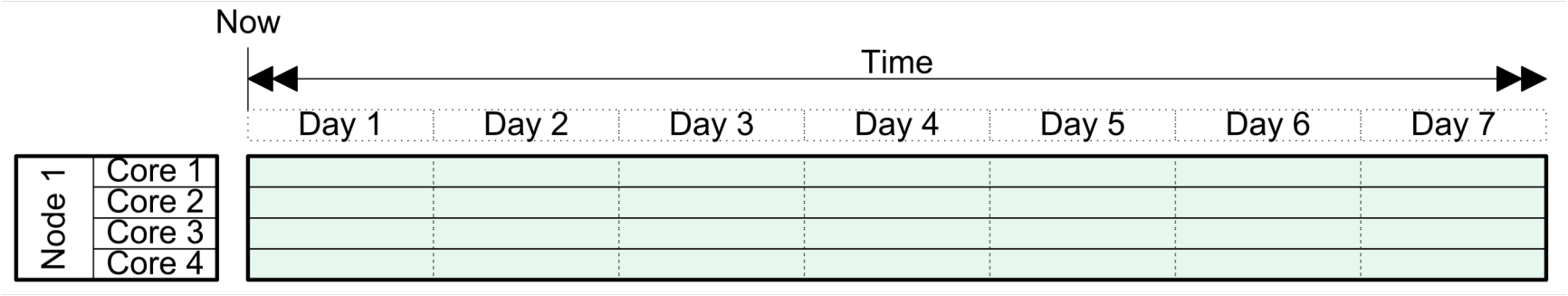
Typical HPC Cluster



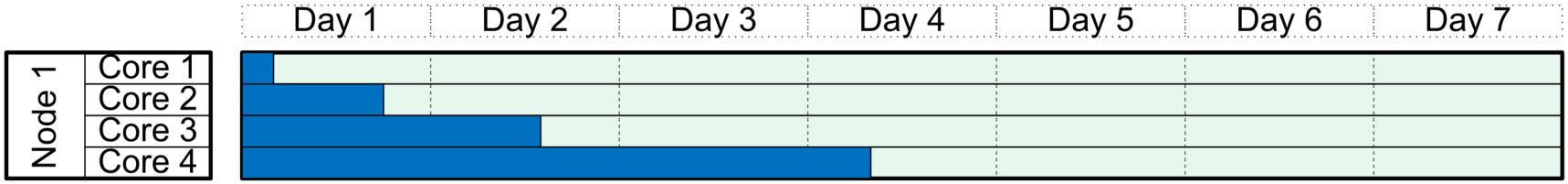
Bigger HPC Cluster



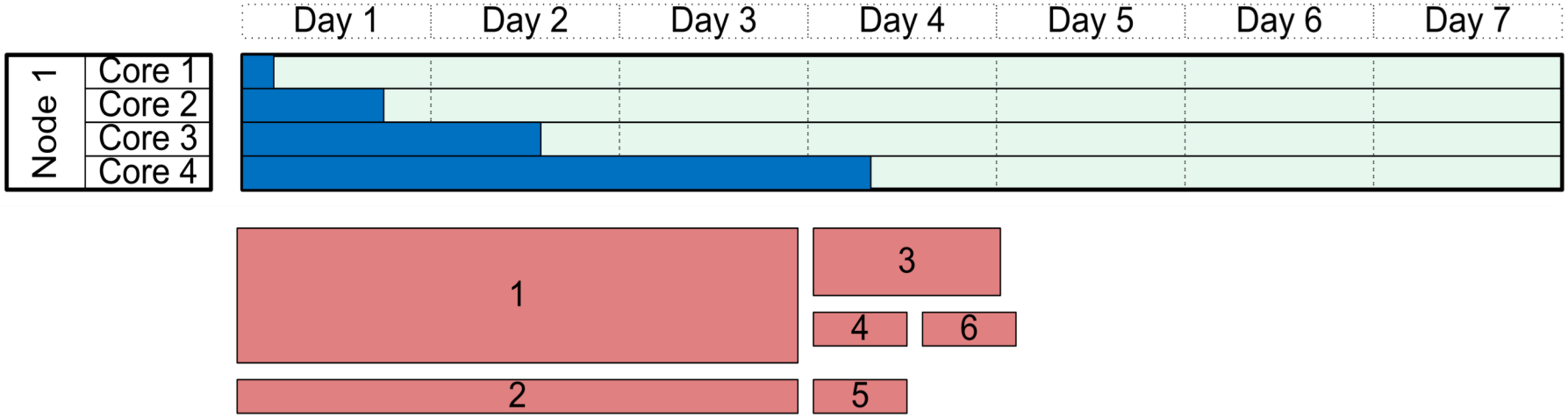
Visualizing single node cluster



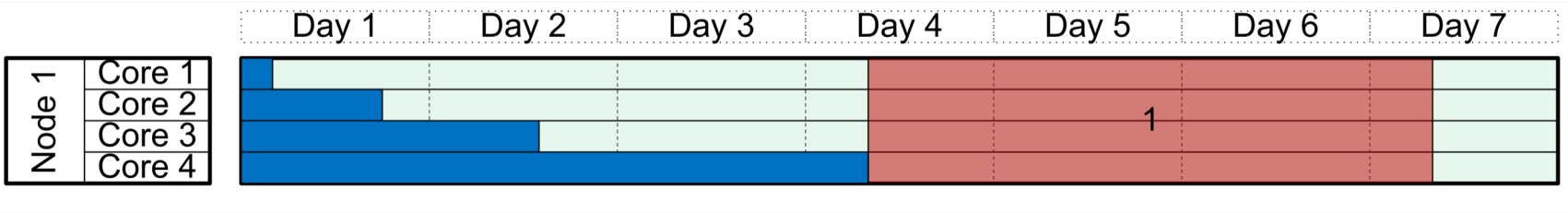
Running jobs



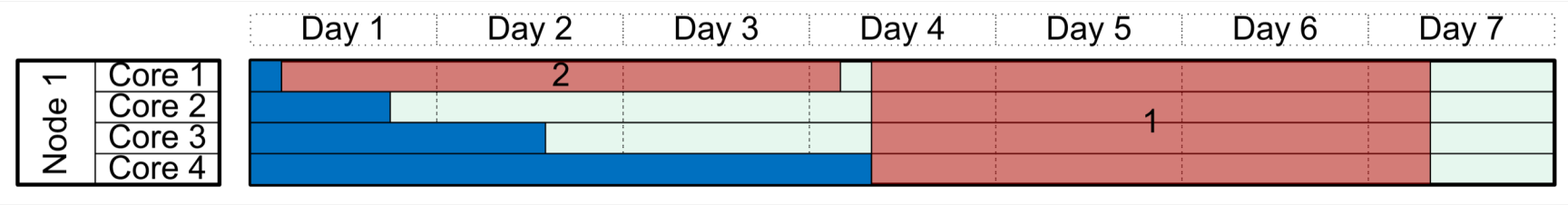
Scheduling jobs in order of priority



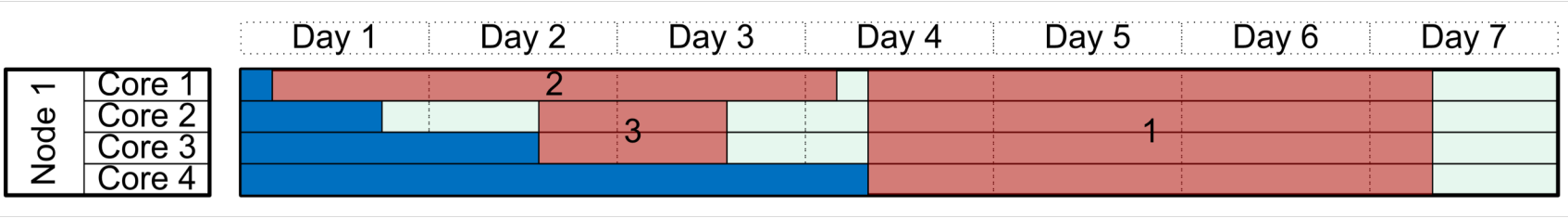
Scheduling jobs in order of priority



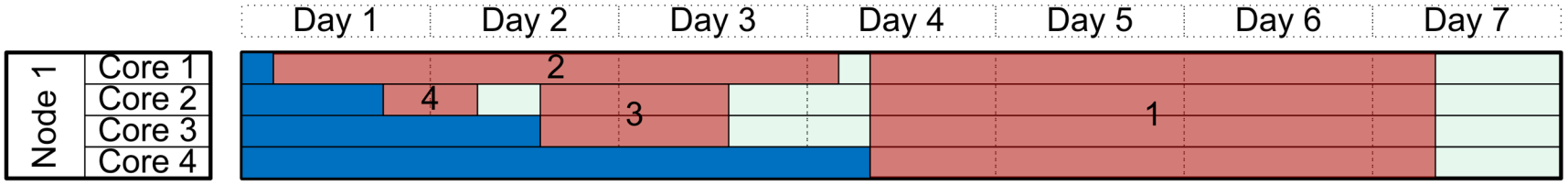
Scheduling jobs in order of priority



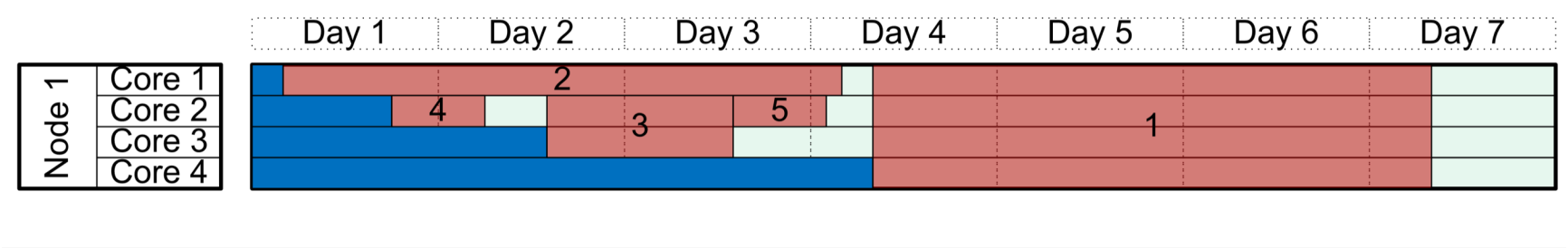
Scheduling jobs in order of priority



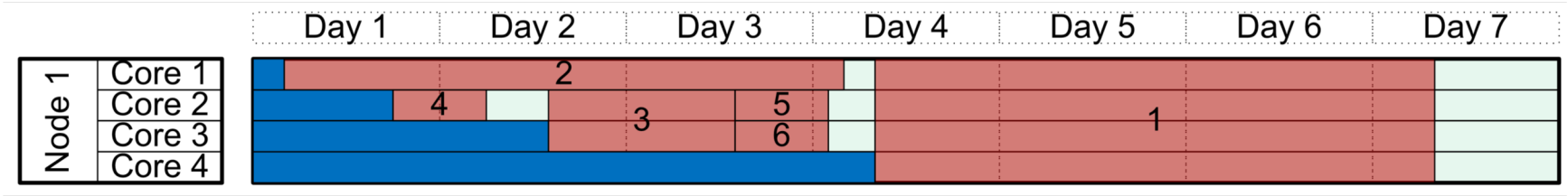
Scheduling jobs in order of priority



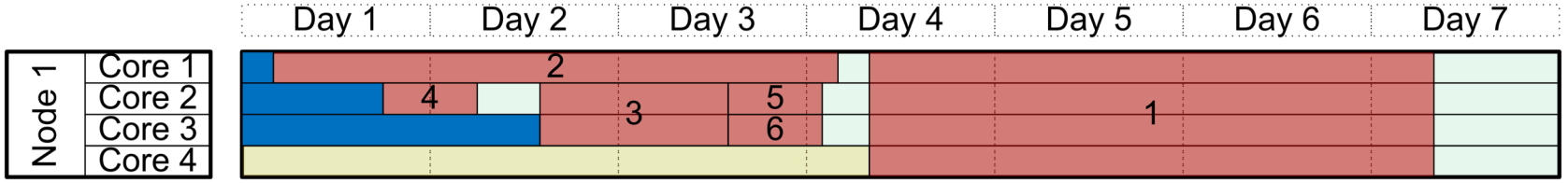
Scheduling jobs in order of priority



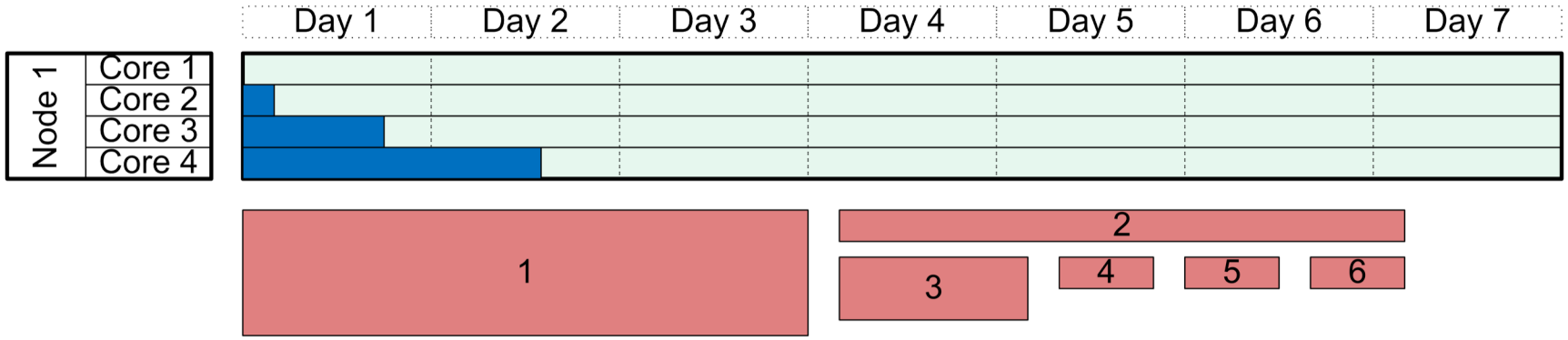
Scheduling jobs in order of priority



A Job finishes early

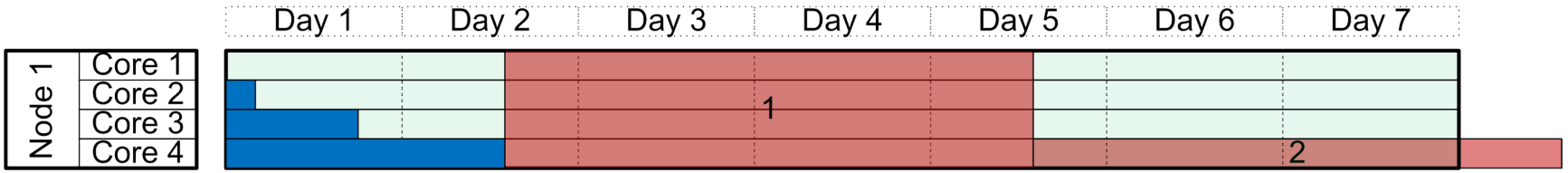


Jobs are rescheduled

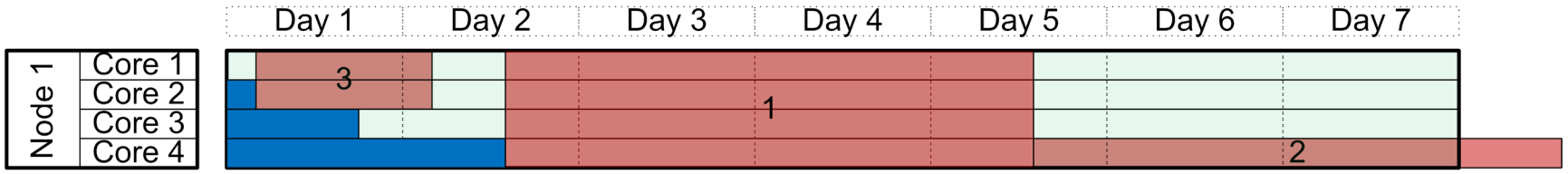


Jobs are rescheduled

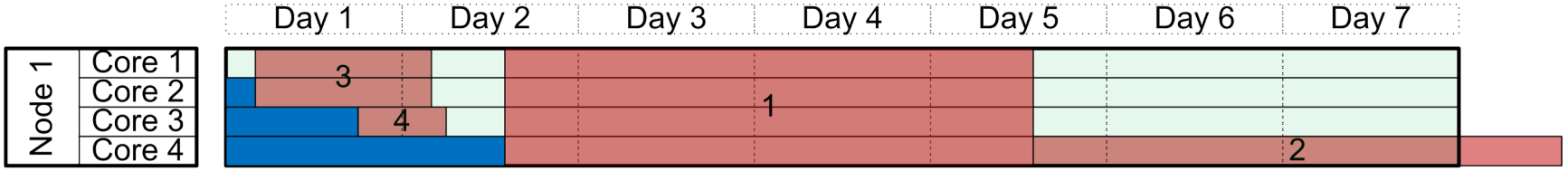
Jobs are rescheduled



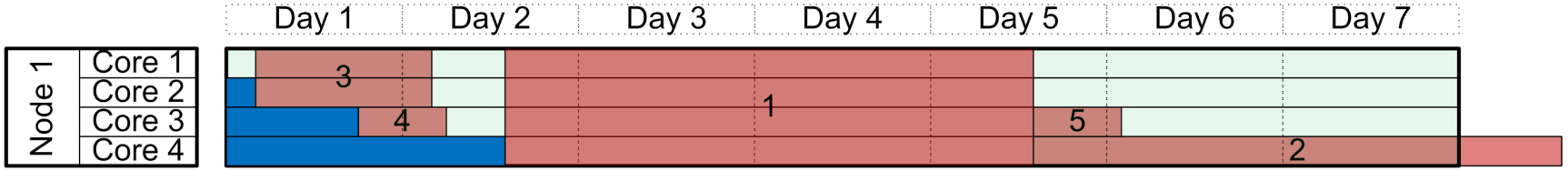
Jobs are rescheduled



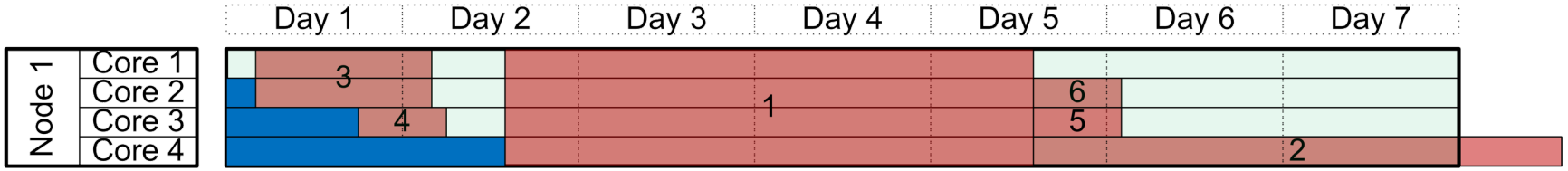
Jobs are rescheduled



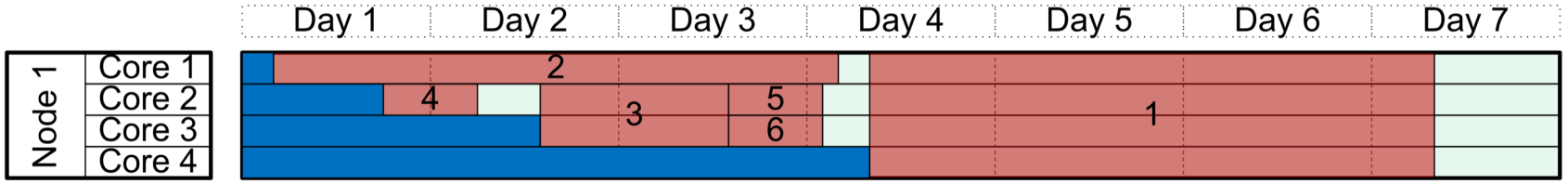
Jobs are rescheduled



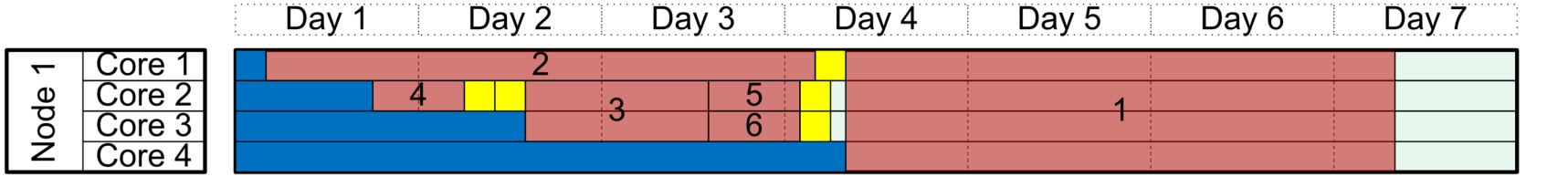
Jobs are rescheduled



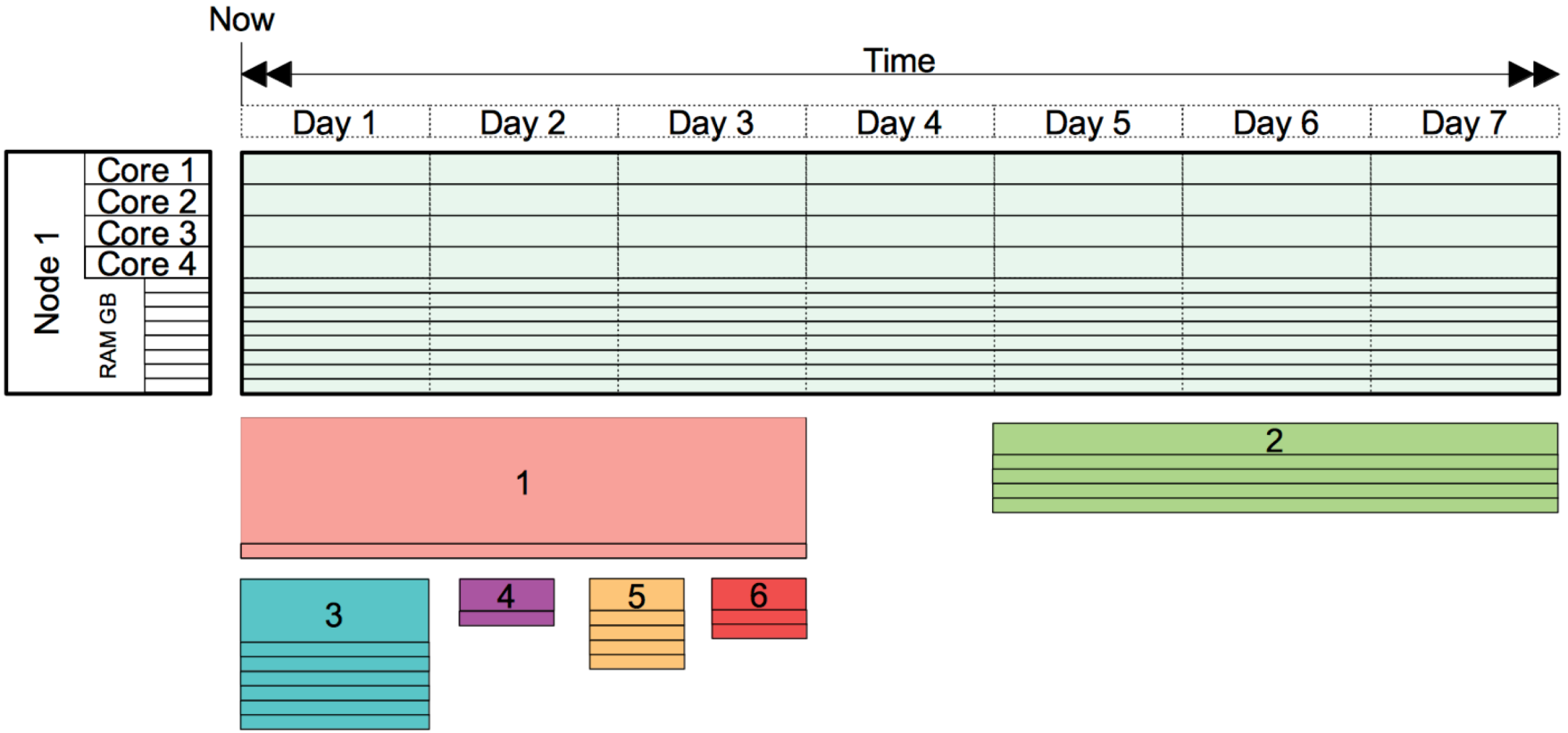
Single node cluster



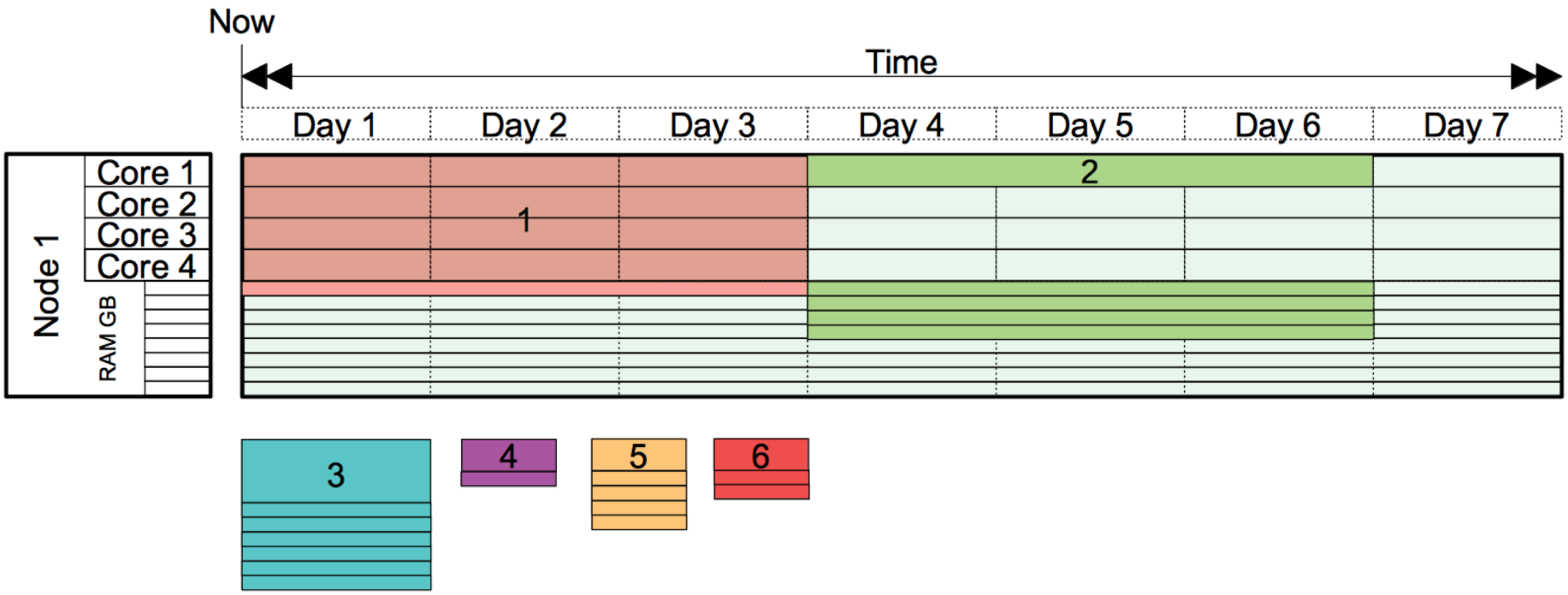
Short serial jobs and Backfill



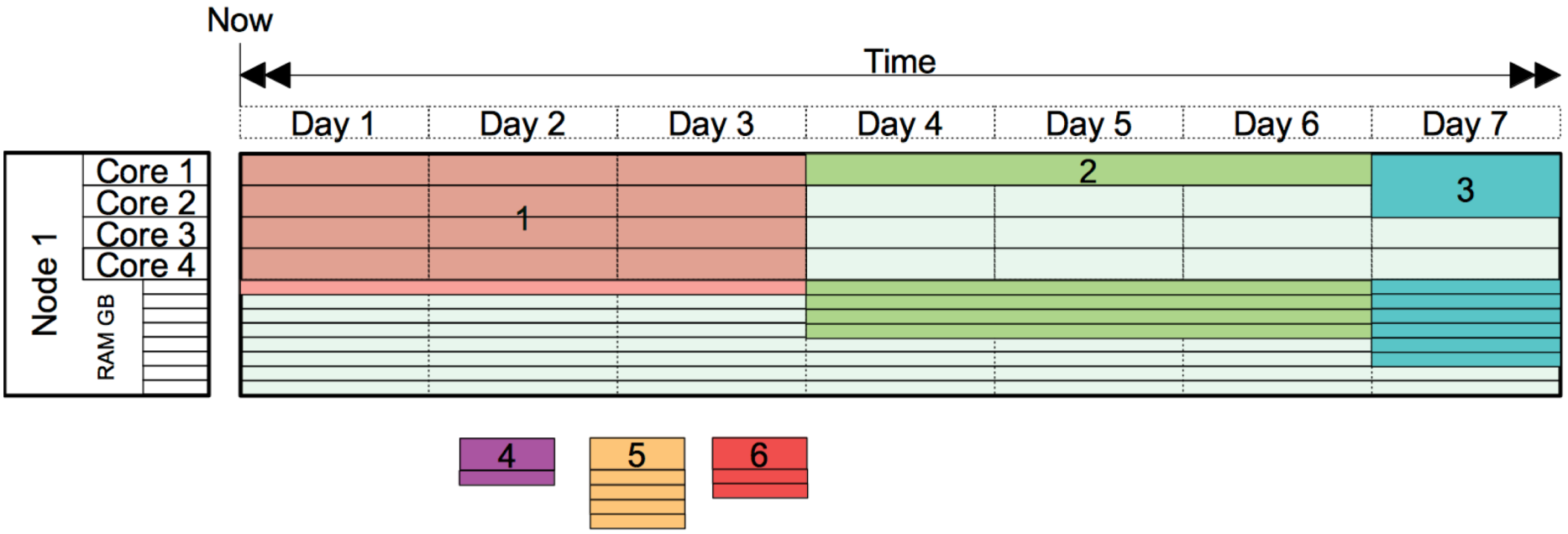
Scheduling Cores and Memory



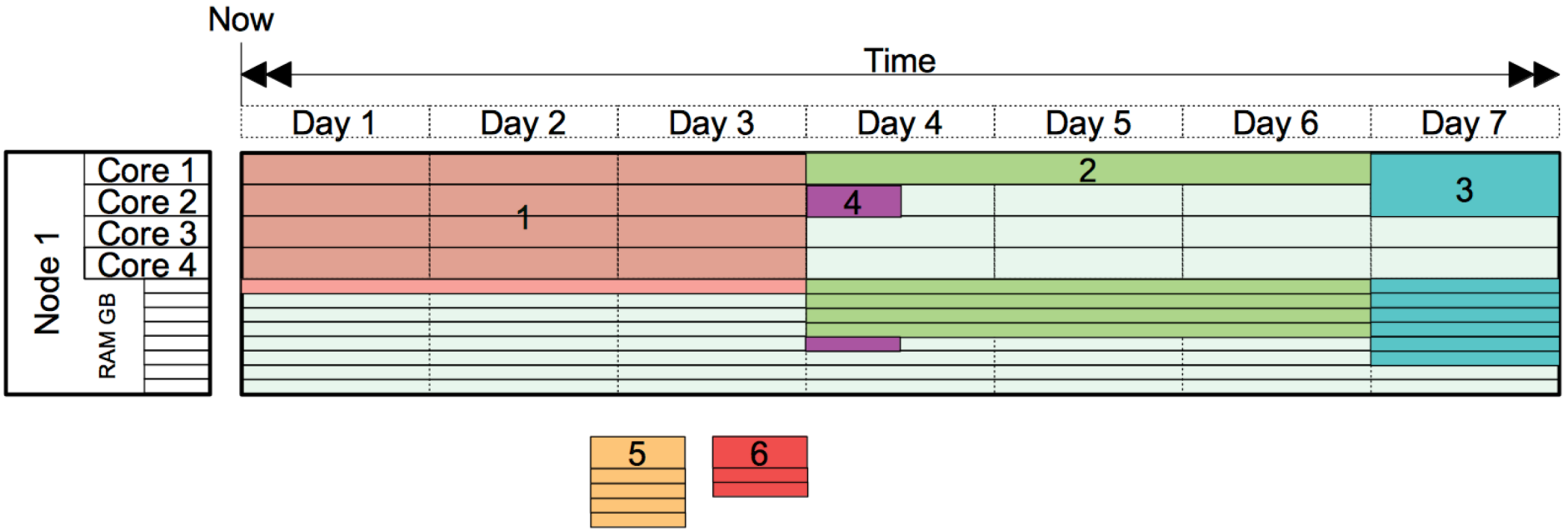
Scheduling Cores and Memory



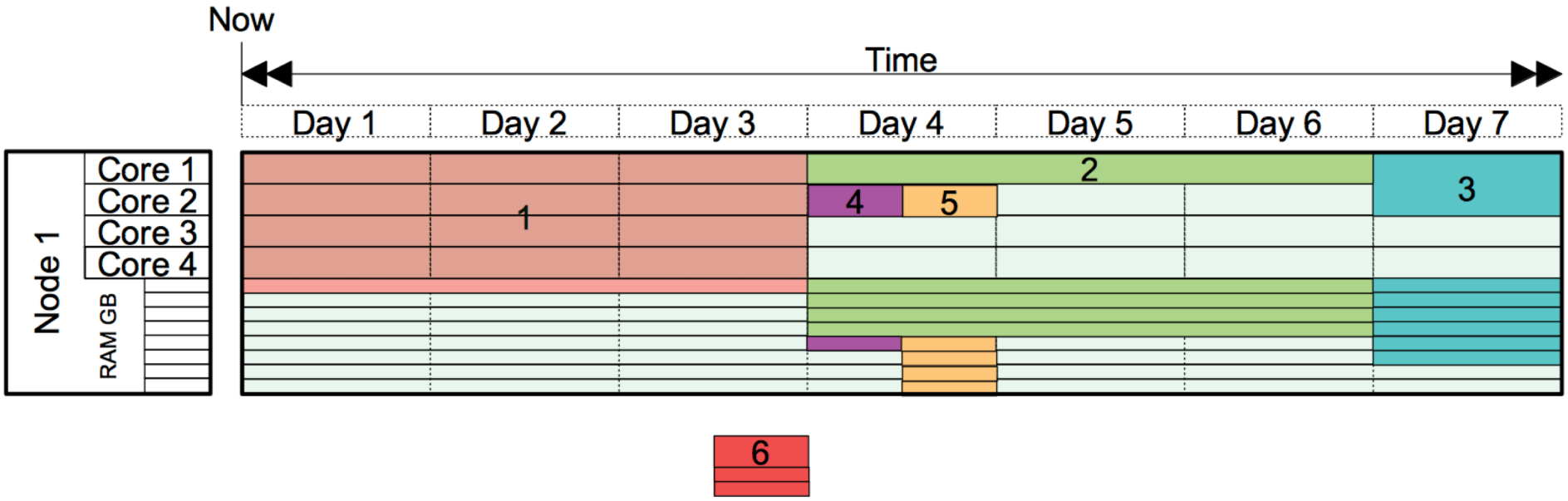
Scheduling Cores and Memory



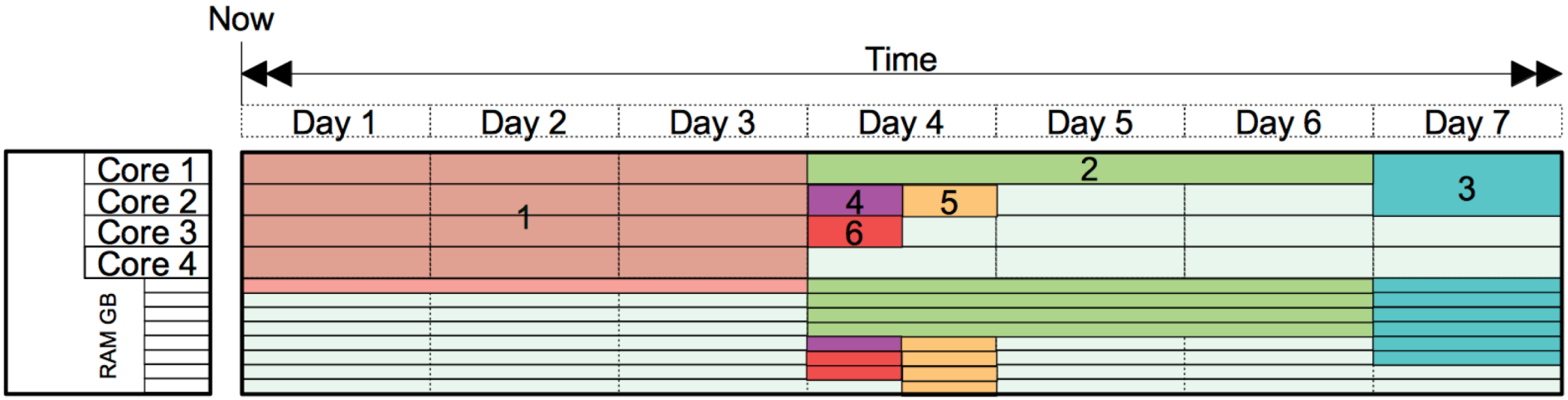
Scheduling Cores and Memory



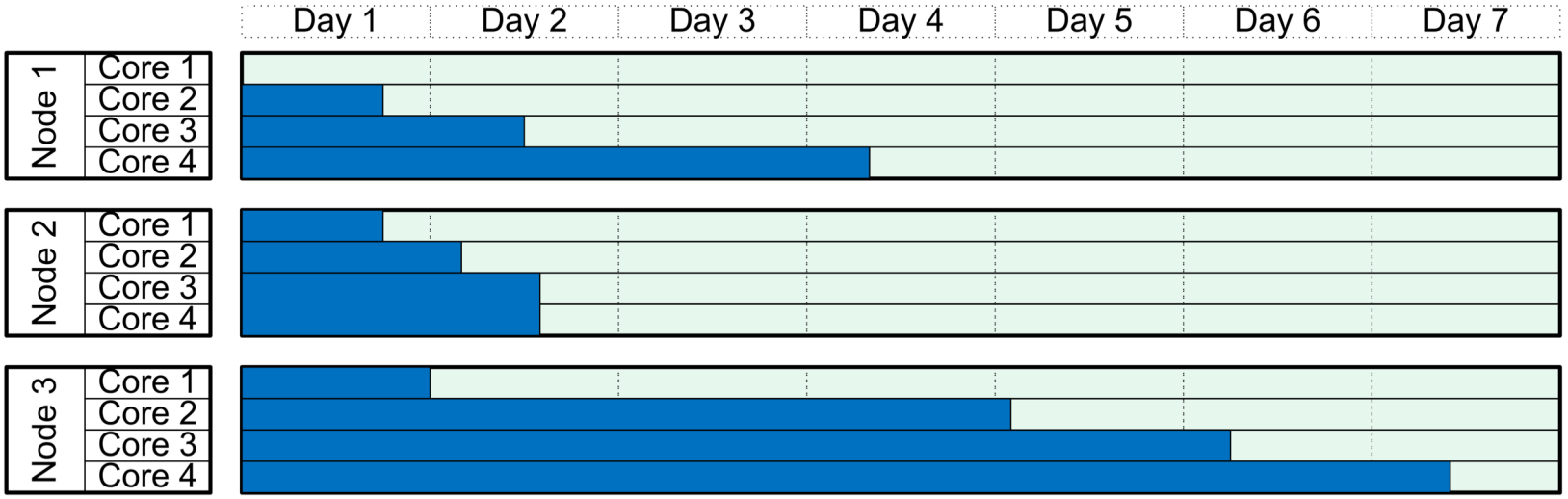
Scheduling Cores and Memory



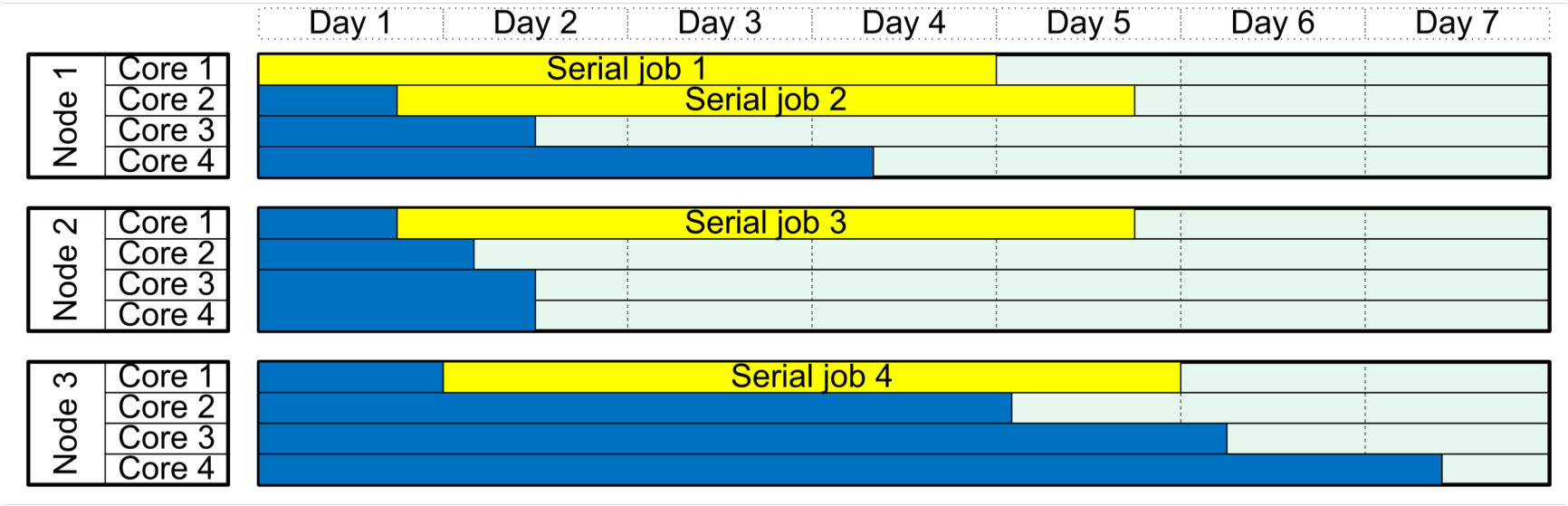
Scheduling Cores and Memory



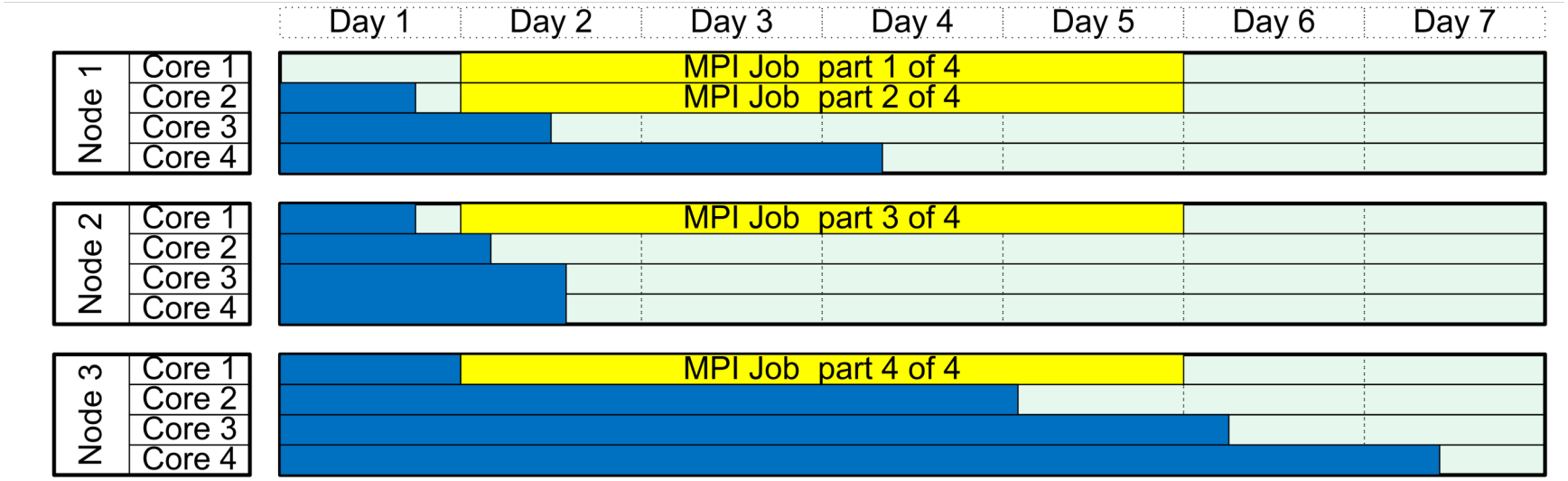
Visualizing Multinode cluster



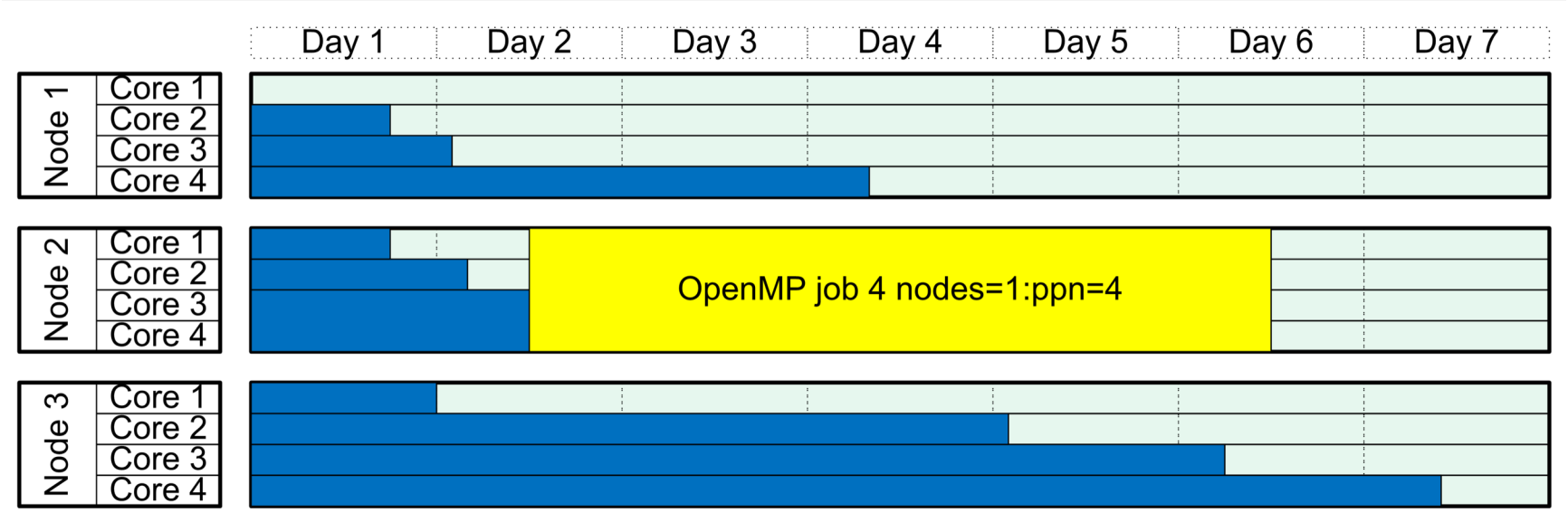
Many Serial Jobs



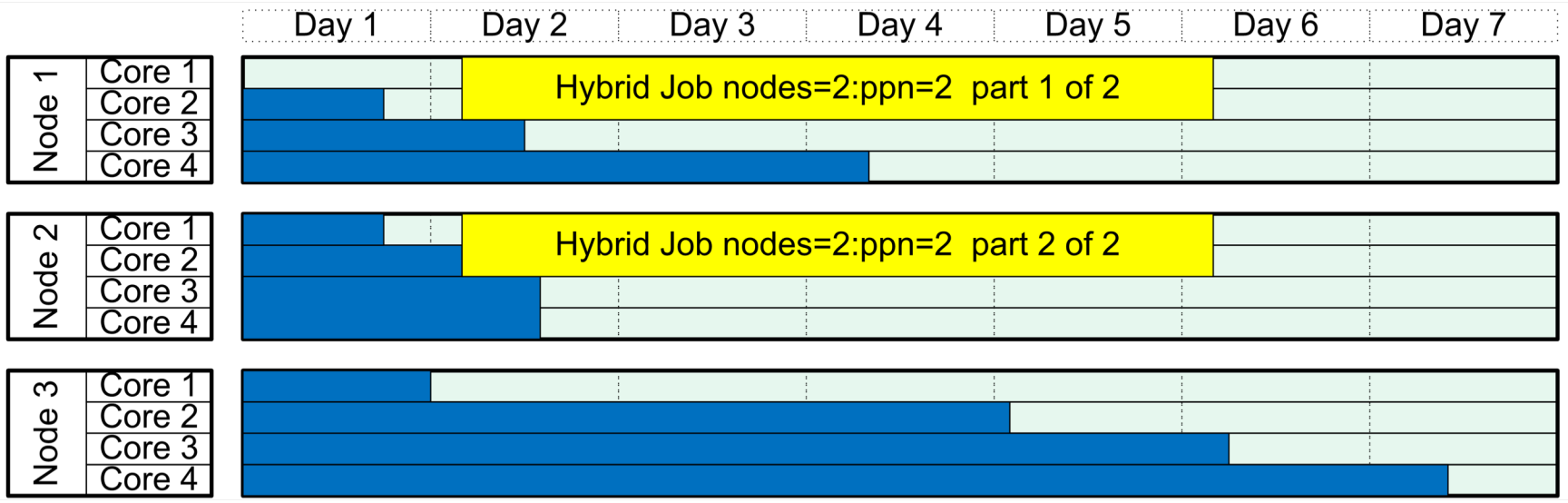
MPI job


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