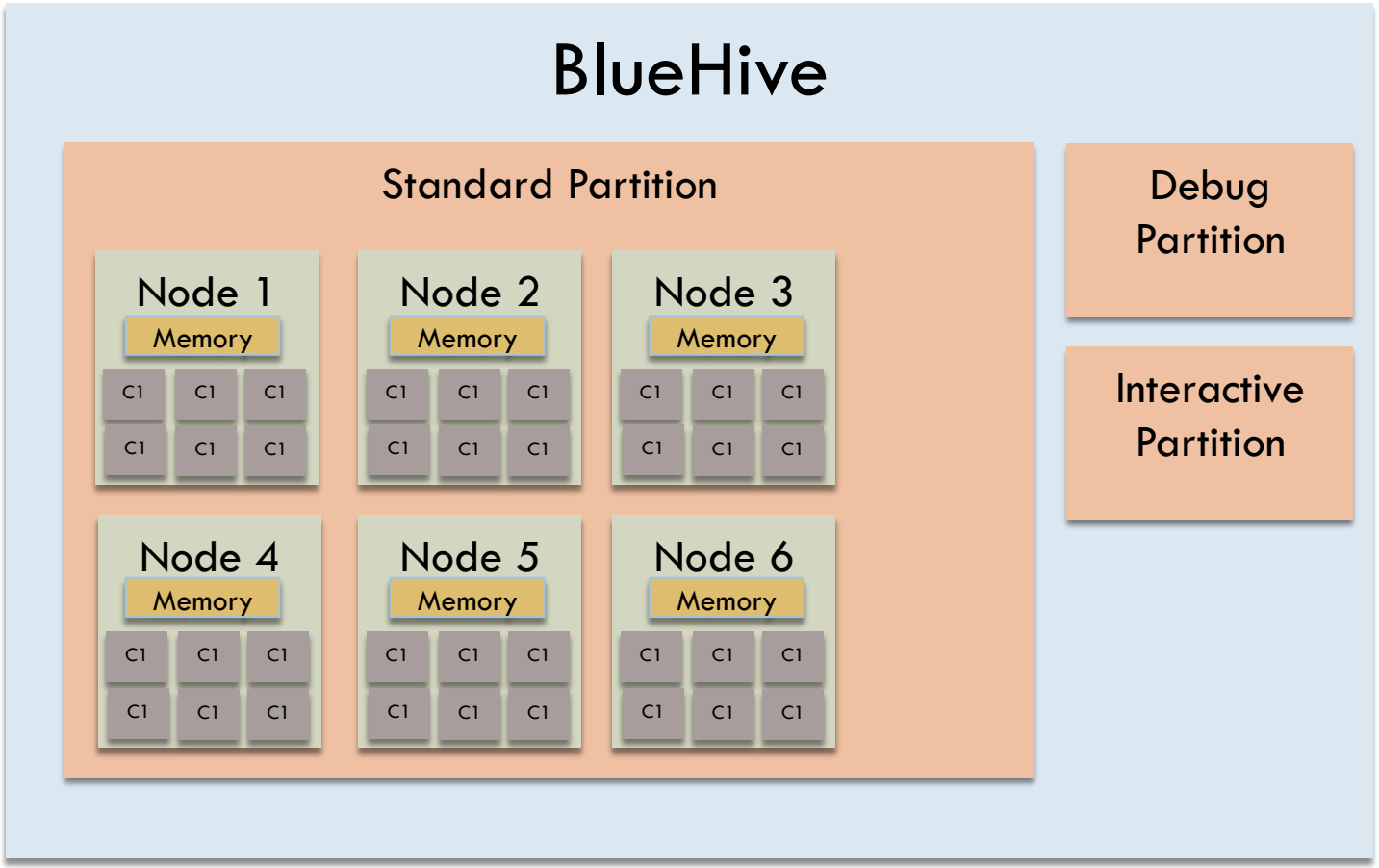

Simple Linux Utility for Research Management

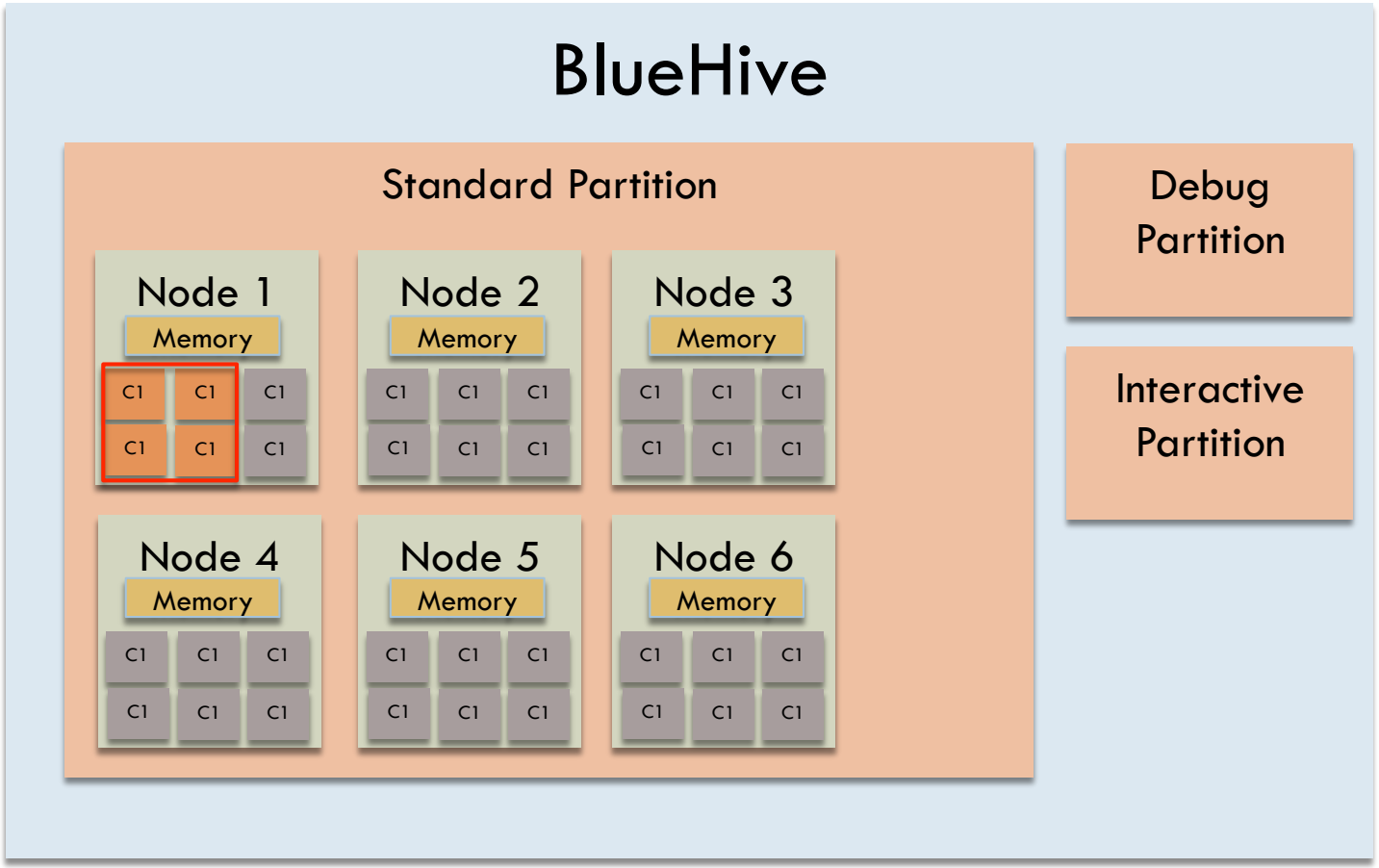
# Outline

- Types of Jobs and job resources
  - walltime, cpus, memory, accelerators
- Running Jobs
  - interactive
  - srun
  - salloc
  - sbatch
- Example Scripts
  - MPI Jobs
  - OpenMP Jobs
  - Hybrid Jobs
  - Job Arrays
- X2Go Desktop
- Remove Visualization

# Job Resources

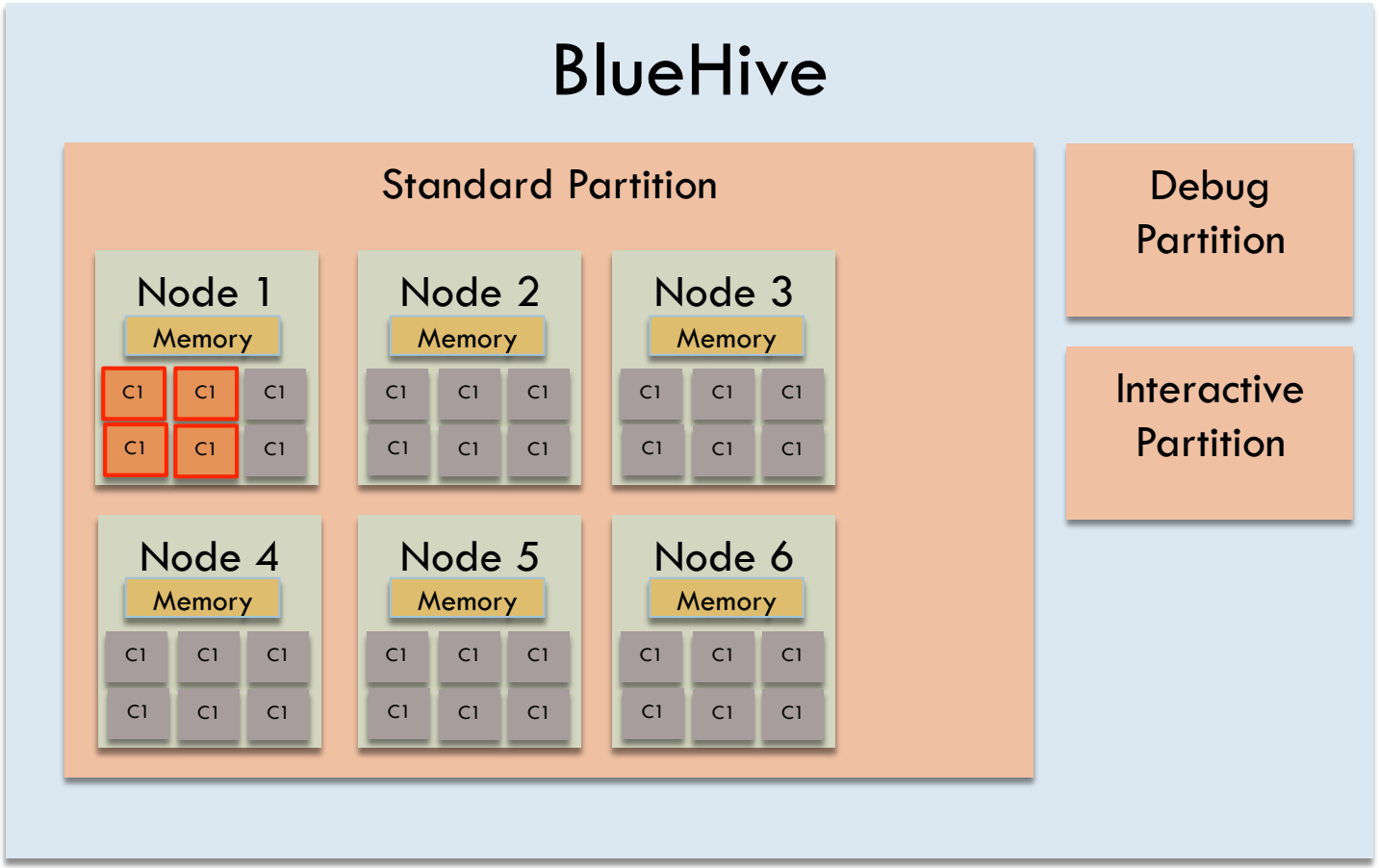


# Job Resources



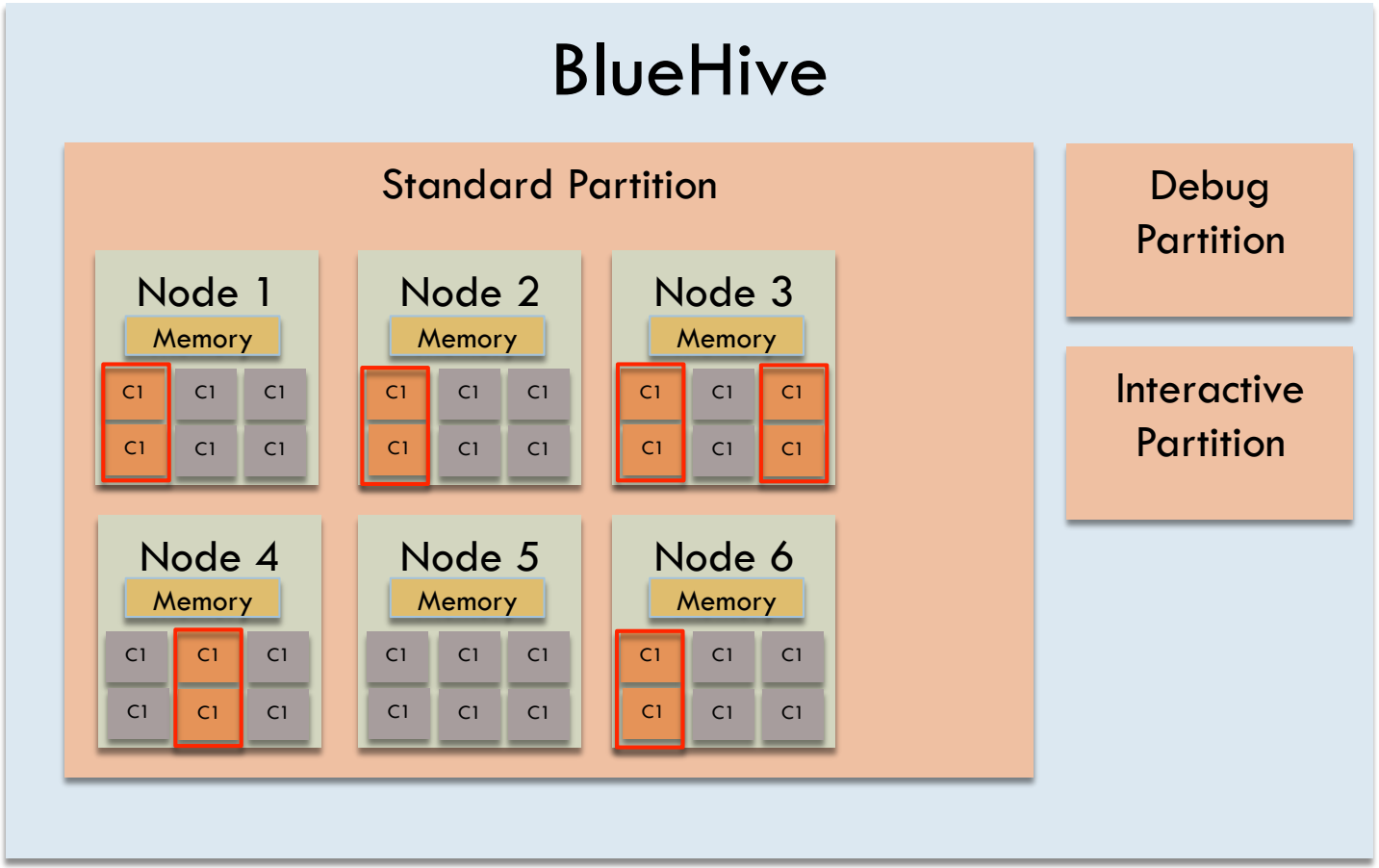

1 task that uses 4 cores – (Shared memory parallelism)

# Job Resources



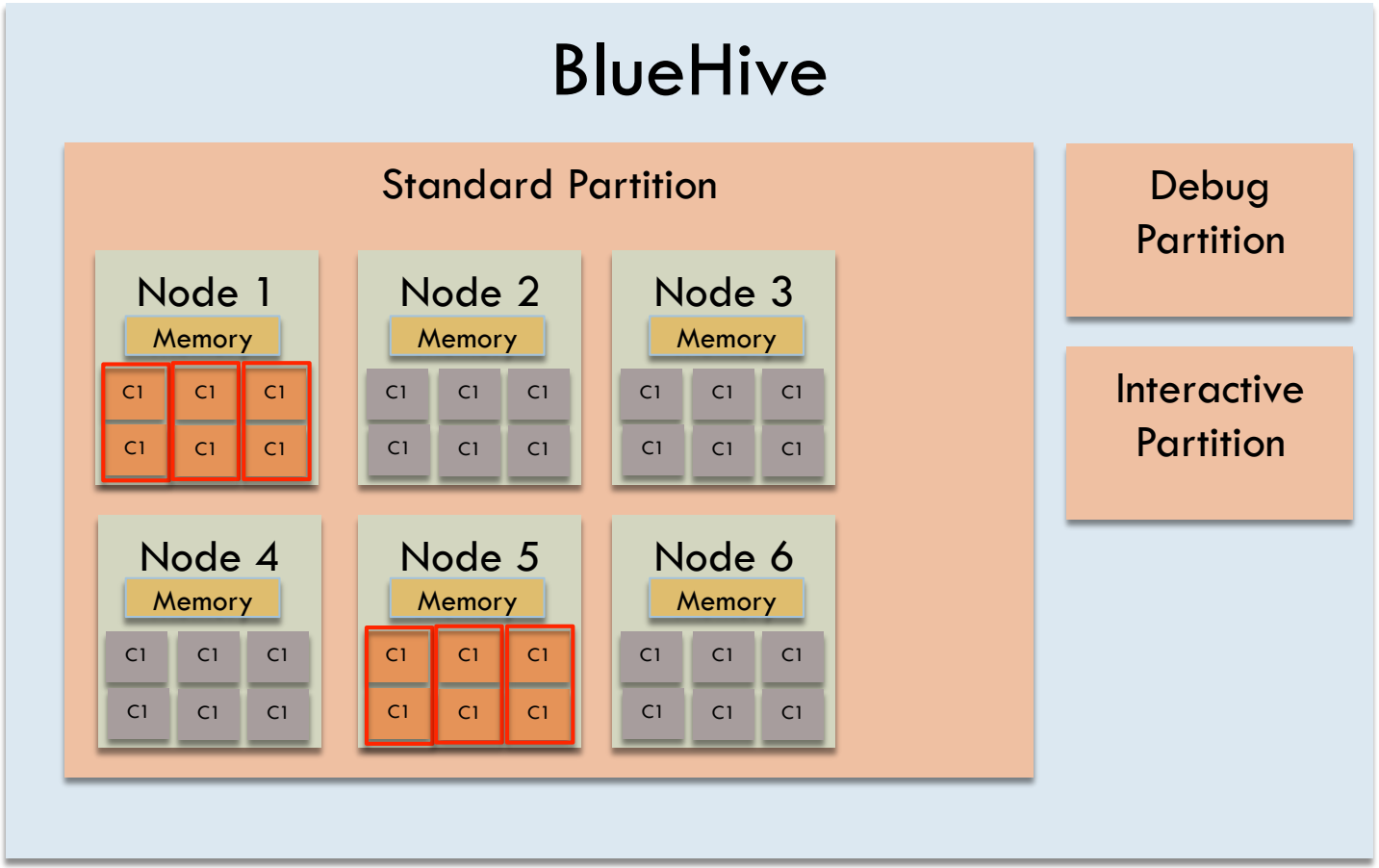

4 tasks that uses 1 core on the same node – (Distributed memory parallelism)

# Job Resources



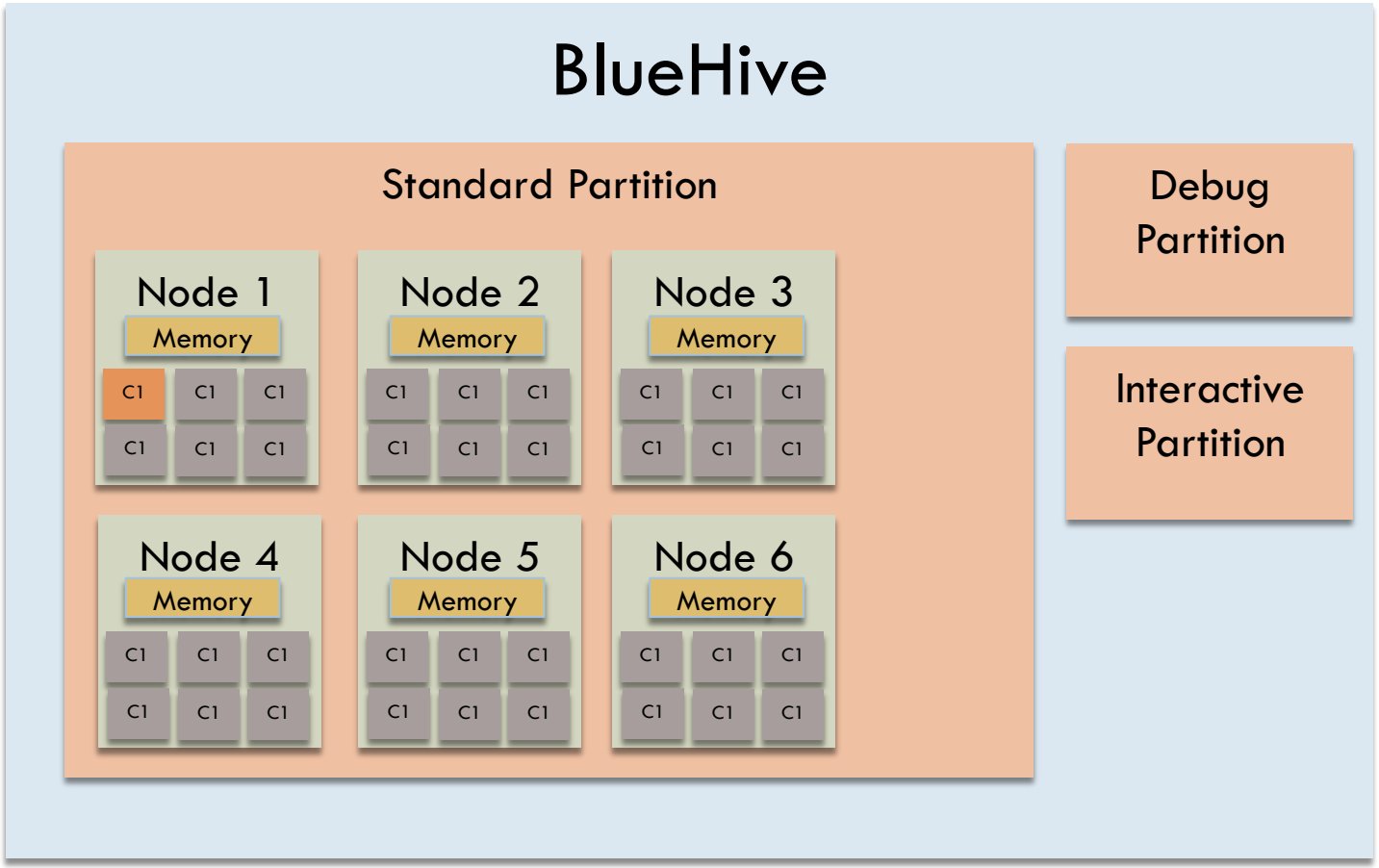

6 tasks that use 2 cores on different nodes – (Hybrid = Distributed + Shared)

# Job Resources



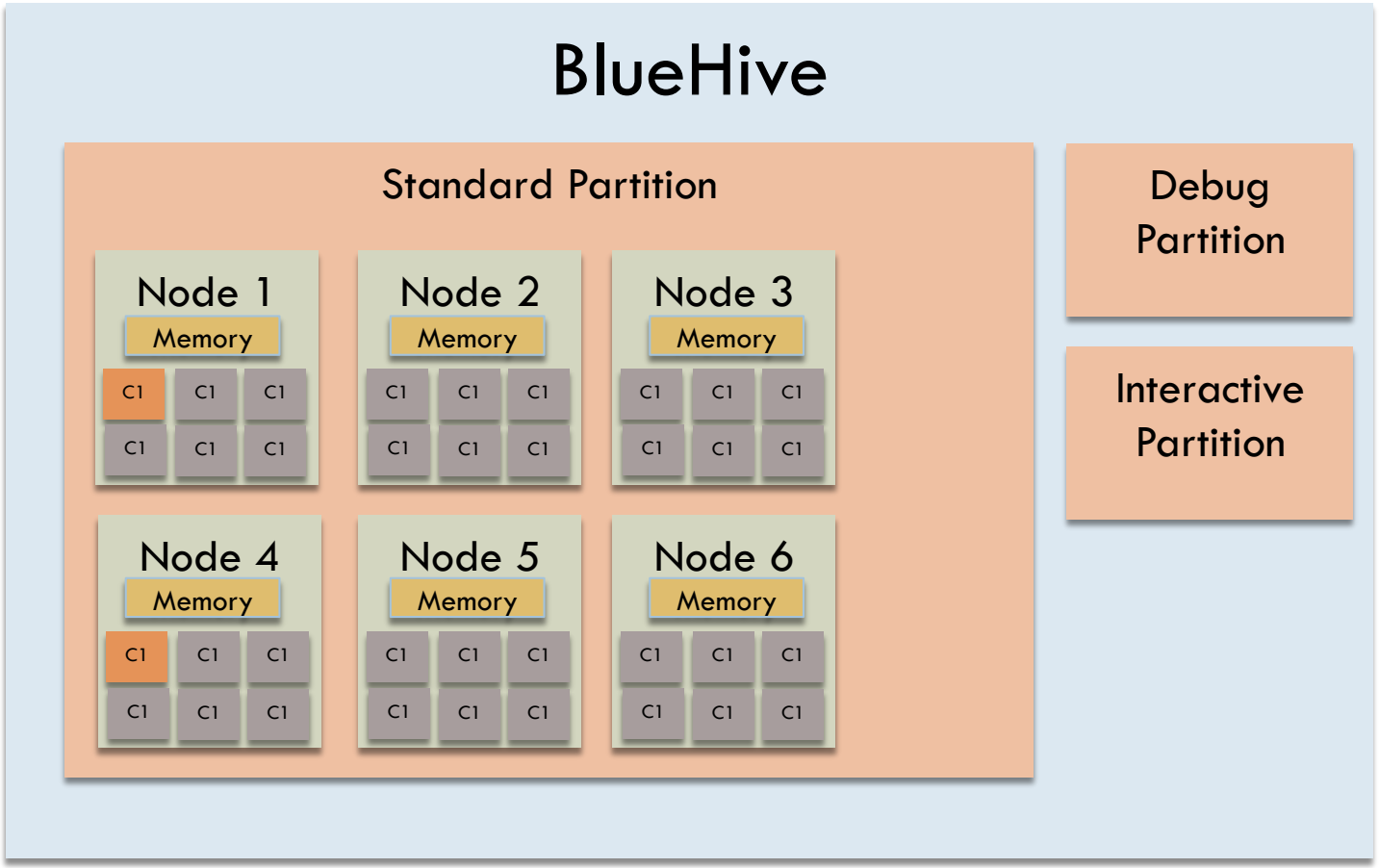

6 task that use 12 cores on 2 nodes – (Hybrid)

# Job Resources



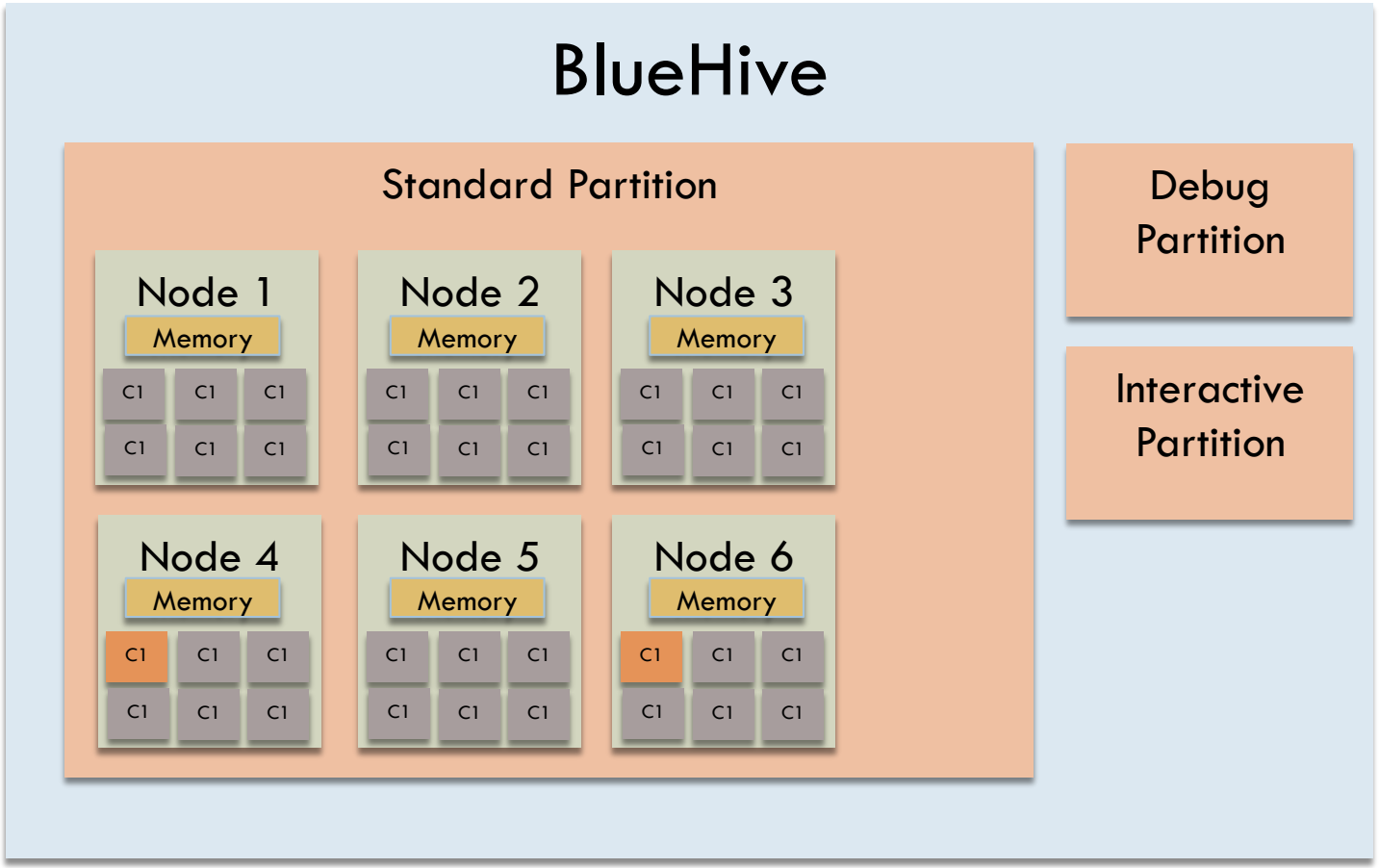

Lots of independent jobs that use 1 task and 1 core (Job Array)

# Job Resources




Lots of independent jobs that use 1 task and 1 core (Job Array)

# Job Resources




Lots of independent jobs that use 1 task and 1 core (Job Array)

# Running Jobs – continued

- Basic Job options:
  - --partition=debug — Partition
  - --time=1-00:00:00 — Wall Time (days-hours:minutes:seconds)
  - --cpus-per-task=1 — Number of cpus per task

  - --ntasks=4 — Number of tasks
  - -mem-per-cpu=2gb — Memory per cpu

  - --nodes=2 — Number of nodes.
  - --ntasks-per-node=12 — Tasks per node
  - ---mem=32gb — Memory per node

  - --array=1,2,3-5 — Create an array of jobs

# Running Jobs – continued

- Additional job parameters:

| | |
|---|---|
| --gres=gpu:1 | Requests gpus |
| --gres=mic:1 | Request intel phi cards |
| --job-name=my_job | Name of your job |
| --output=my_output.%j | Standard output |
| --error=my_err.%j | Standard error |
| --mail-type=begin | When to send e-mails pertaining to your job. [begin, end, fail, requeue, or all] |
| --mail-user=<email> | Use specific email address |
| --account=jcarrol5_lab | Specify a specific account to use (only for user owned hardware) |

# Running Jobs

- Several of the more common options have shortened versions
  - --time=60 or -t 60
  - --cpus-per-task=2 or -c 2
  - --nodes=4 or -N 4
  - --ntasks=2 or -n 2
  - --job-name=MyJob or -J MyJob
  - --partition=debug or -p debug

# Running Jobs

- interactive – Creates a job and gives you a login prompt on the terminal of the first node in your job.
  - interactive -p debug -n 4 -t 2
- srun – executes an instruction across all of the tasks associated with the job.
  - srun hostname (runs 4 copies of 'hostname')
- salloc – similar to interactive except that it leaves you on the login node – but in a subshell
  - salloc -p debug -n 4 -t 2
  - exit

# Running Jobs – continued

- sbatch – submits a script to be run at a later time as a batch job. For example:
  - sbatch myjob.slurm

myjob.slurm

```
#!/bin/bash
#SBATCH -N 2
#SBATCH --ntasks-per-node 24
srun my_program
srun my_other_program
```

  - sbatch can take command line arguments. For example
    - sbatch -N 3 myjob.slurm
  - sbatch can also read from the standard in

# Running Jobs – continued

- srun --pty $SHELL will give you a terminal, but will not forward your X display. To do this, you will need to use the interactive script
  - interactive

- JobLauncher - GUI that can be used to submit interactive jobs.

# Running Jobs – continued

- When you use any of these commands, your environment is transferred to the nodes your job runs on. So the following will work fine.
  - module load matlab
  - salloc
  - srun matlab
- Or you explicitly load your modules in your sbatch script.

# MPI Jobs

- Sample MPI Job
  - #!/bin/bash
  - #SBATCH -t 1-00:00:00
  - #SBATCH -N 4
  - #SBATCH --ntasks-per-node 24
  - #SBATCH --mem-per-cpu 2000
  - mpirun my_mpi_program

- Asks for 4 nodes with 24 tasks per node for 1 day. For most mpi libraries, you can use 'srun' instead of 'mpirun'.

# OpenMP Jobs

- Sample OpenMP Job
  - #!/bin/bash
  - #SBATCH -t 1-00:00:00
  - #SBATCH -n 1
  - #SBATCH --cpus-per-task 12
  - #SBATCH --mem-per-cpu 2000
  - srun my_openmp_program

- Asks for 1 task with 12 cpus.

# Hybrid Jobs

- Sample Hybrid (MPI-OpenMP) Job
  - #!/bin/bash
  - #SBATCH -t 1-00:00:00
  - #SBATCH -N 3
  - #SBATCH --ntasks-per-node 4
  - #SBATCH --cpus-per-task 6
  - #SBATCH --mem-per-cpu 2000
  - mpirun my_hybrid_program

- Asks for 3 nodes with 4 tasks per node with 6 cpus per task. If we didn't care whether our 12 mpi tasks were distributed across more than 3 nodes, we could also have set "-n 12" and not set "-N" or "--ntasks-per-node"

# Job Arrays

- Sample Job Array
  - #!/bin/bash
  - #SBATCH --array=1-3
  - #SBATCH -t 1-00:00:00
  - #SBATCH -n 1
  - #SBATCH --mem-per-cpu 2000
  - srun my_program $SLURM_ARRAY_TASK_ID

# Other useful slurm commands

- sinfo - displays information about the various queue limits
- squeue - displays information about jobs in the queue
- scancel - kills any submitted jobs

# X2go, JobLauncher, and Remote Vis

- You can launch some applications directly from X2go's menu using the JobLauncher utility.

- Some visualization applications will benefit from having an OpenGL capable GPU on board. This involves running a remote desktop session (similar to X2go), but directly on a compute node in the visual partition.

  https://info.circ.rochester.edu/BlueHive/Remote_Visualization.html

# Questions

- For more information see
  - https://info.circ.rochester.edu/BlueHive/Running_Jobs.html
  - man bluehive
  - man sbatch