OAK RIDGE National Laboratory

Burst Buffer on Summit

George S. Markomanolis,
HPC Engineer
Oak Ridge National Laboratory

Summit New User Training

Knoxville, TN

3 June 2020

ORNL is managed by UT-Battelle, LLC for the US Department of Energy

U.S. DEPARTMENT OF ENERGY

Burst Buffer on compute node

- Burst Buffers are technologies that provide faster I/O based on new media, on Summit we have on each compute node a Samsung PM1725a NVMe
- 4,608 nodes with local NVMe of 1.6 TB
  - 7.3 PB Total
  - Write performance per BB node: 2.1 GB/s
  - Read performance per BB node : 5.5 GB/s
- By default we can do one file per MPI process or one file per node, no single shared file between different Burst Buffer nodes without using any other Burst Buffer library (check Chris Zimmer's presentation tomorrow).
- Linear scalability by using Burst Buffers across many nodes
- Exclusive usage of the resources, no sharing with other users

Burst Buffer – Use cases

- Periodic burst
- Good for machine learning and deep learning workloads
- Transfer to PFS between bursts
- I/O improvements
- Improves applications with heavy metadata

Open slide master to edit

Burst Buffer

- Burst Buffer can be used through the scheduler, integration with LSF
- What a user has to do?
  - Add the appropriate scheduler option in the submission script
  - Copy any necessary file (if required) on the Burst Buffer (input file, executable)
  - Execute the application and make sure that it reads/writes the files with significant size from/on Burst Buffer
  - Copy the necessary files back to GPFS to save them

Open slide master to edit

Submission script for Burst Buffer – NAS BTIO

GPFS

```
#!/bin/bash
#BSUB -P projid
#BSUB -J nas_btio
#BSUB -o nas_btio.o%J
#BSUB -W 10
#BSUB -nnodes 1

jsrun -n 1 -a 16 -c 16 -r 1 ./btio
```

Burst Buffer

```
#!/bin/bash
#BSUB -P projid
#BSUB -J nas_btio
#BSUB -o nas_btio.o%J
#BSUB -W 10
#BSUB -alloc_flags "nvme"
#BSUB -nnodes 1

jsrun -n 1 cp btio  inputbt.data /mnt/bb/$USER/

jsrun -n 1 -a 16 -c 16 -r 1 /mnt/bb/$USER/btio

jsrun -n 1 cp /mnt/bb/$USER/btio.nc
/gpfs/alpine/scratch/…
```


Open slide master to edit

NAS BTIO

- Executing 16 MPI processes on a **single** BB node, blocking PNetCDF with a single shared file

Total I/O amount	:	152.6 GB
Time in sec	:	67.98
I/O bandwidth	:	2.24 GB/s

Open slide master to edit

Understanding the MPI I/O Hints

- Using the command *export ROMIO_PRINT_HINTS=1* in the submission script, we can acquire the following information for 16 MPI processes of one BB node

```
key = cb_buffer_size          value = 16777216
key = romio_cb_read             value = enable
key = romio_cb_write            value = enable
key = cb_nodes                  value = 1
…
key = cb_config_list            value = *:1
key = romio_aggregator_list     value = 0
```


Open slide master to edit

NAS BTIO - Improved

- Increasing the MPI I/O aggregators to 8

```
echo "cb_config_list *:8" > romio_hints
```

- Declare the ROMIO_HINTS variable

```
export ROMIO_HINTS=$PWD/romio_hints
```

- New performance results

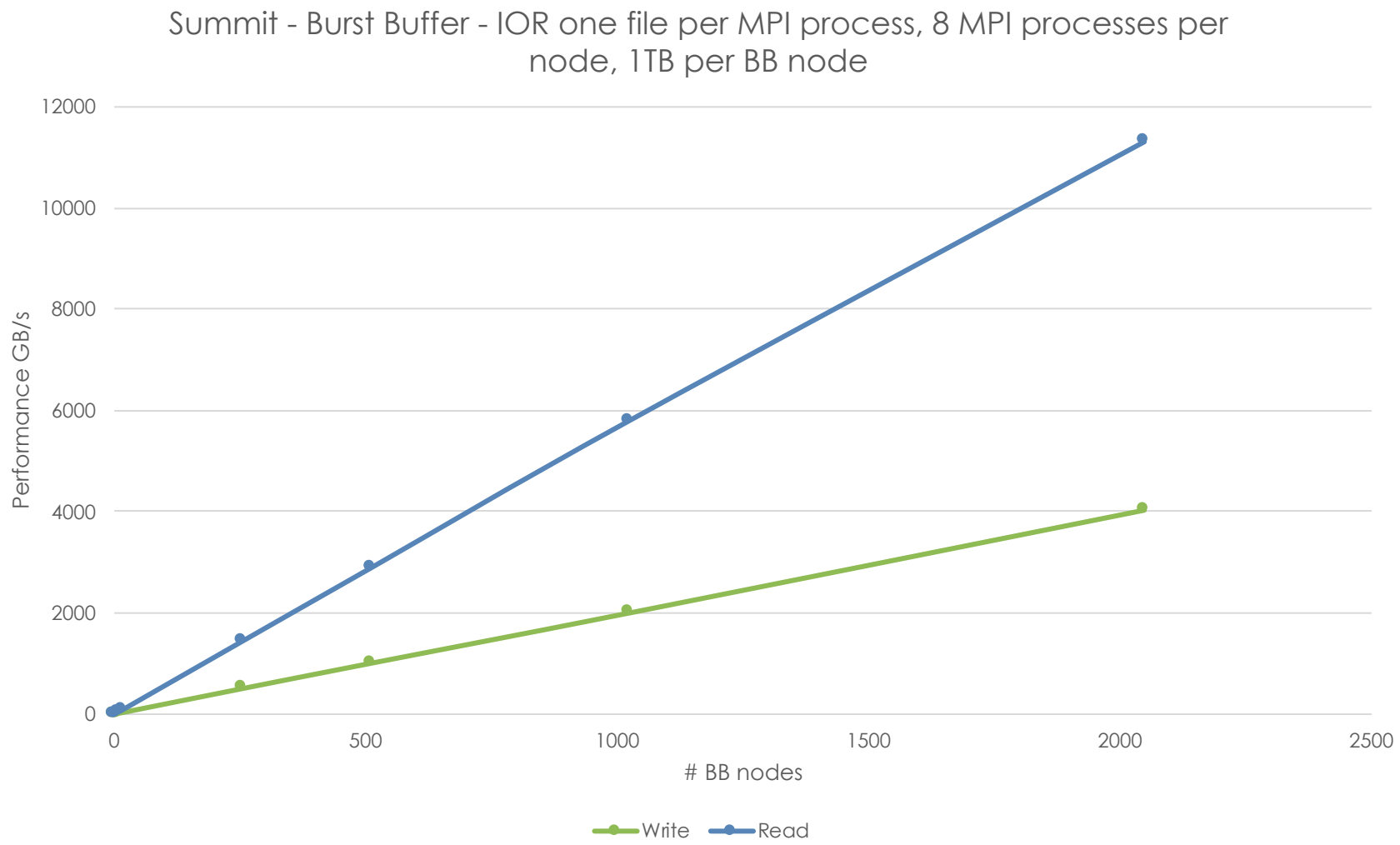
```
Totail I/O amount      :    152.6 GB
Time in sec            :     52.47
I/O bandwidth          :     2.98 GB/s
```

Almost 23% improvement by using page cache and NVMe

Open slide master to edit

Burst Buffer

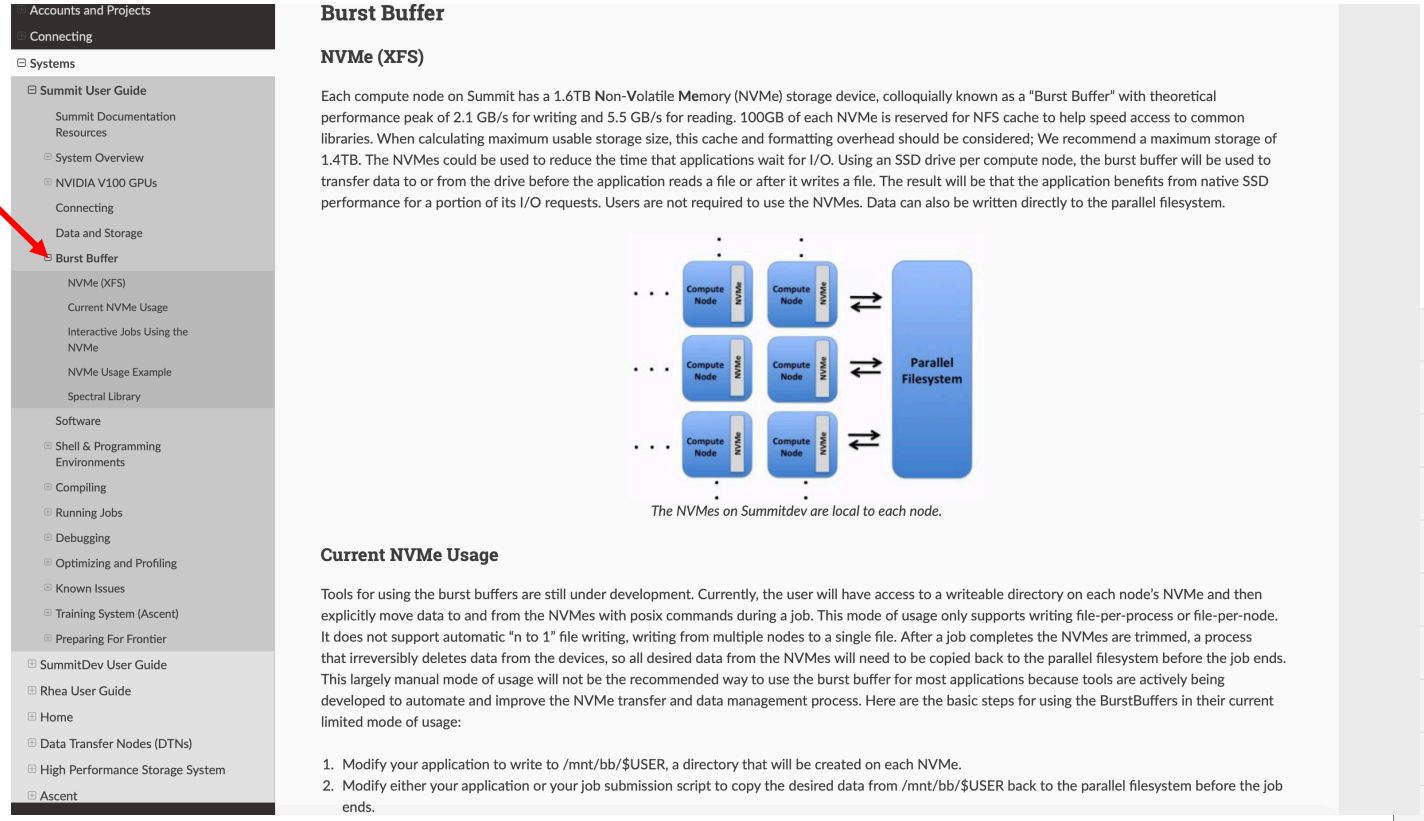
- Scalability test with IOR



OAK RIDGE National Laboratory

Open slide master to edit

Documentation



Using GPFS	*Using NVMe*
`#!/bin/bash`	
`#BSUB -P xxx`	
`#BSUB -J NAS-BTIO`	
`#BSUB -o nasbtio.o%J`	
`#BSUB -e nasbtio.e%J`	
`#BSUB -W 10`	
`#BSUB -nnodes 1`	
	`#BSUB -alloc_flags nvme`
	`export BBPATH=/mnt/bb/$USER/`
	`jsrun -n 1 cp btio ${BBPATH}`
	`jsrun -n 1 cp input* ${BBPATH}`
`jsrun -n 1 -a 16 -c 16 -r 1 ./btio`	`jsrun -n 1 -a 16 -c 16 -r 1 ${BBPATH}/btio`
`ls -l`	`jsrun -n 1 ls -l ${BBPATH}/`
	`jsrun -n 1 cp ${BBPATH}/* .`

https://docs.olcf.ornl.gov/systems/summit_user_guide.html#burst-buffer

OAK RIDGE National Laboratory
Open slide master to edit

Conclusions

- Burst Buffer is the solution for heavy I/O applications
- We need some extra libraries on Summit to support various workflows
- Tuning with MPI I/O hints could provide faster execution time

Open slide master to edit

Thank you!

Questions?

Open slide master to edit