Visualizing Darshan Extended Traces

Jean Luca Bez
Lawrence Berkeley National Laboratory

Jean Luca Bez | jlbez@lbl.gov
Suren Byna | sbyna@lbl.gov

SC’21 BoF: Analyzing Parallel I/O
HPC I/O Stack

- **HPC I/O stack** is complex (multiple layers)
- Interplay of factors can affect I/O performance
- Various **optimizations techniques** available
- Plethora of **tunable parameters**
  - Each layer brings a new set of parameters
- Using the all layers **efficiently** is a **tricky** problem

<table>
<thead>
<tr>
<th>Parallel / Serial Applications</th>
<th>High-Level I/O Libraries</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDFS, NetCDF, ADIOS</td>
<td>POSIX I-O</td>
</tr>
<tr>
<td>OpenMPI, MPICH (ROMIO)</td>
<td>VFS, FUSE</td>
</tr>
<tr>
<td>IBM CIOD, Cray DVS, IOFSL, IOF</td>
<td>I/O Forwarding Layer</td>
</tr>
<tr>
<td>Lustre, GPFS, PVFS2, OrangeFS</td>
<td>Parallel File System</td>
</tr>
<tr>
<td>HDD, SSD, RAID</td>
<td>Storage Devices</td>
</tr>
</tbody>
</table>
Darshan and DXT

- Darshan is a popular tool to collect I/O profiling
- It aggregates information to provide insights
- Extended tracing mode (DXT)

```
export DXT_ENABLE_IO_TRACE=1
```

- Fine grain view of the I/O behavior
- POSIX or MPI-IO, read/write
- Rank, segment, offset, request size
- Start and end timestamp
- How to visualize and extract insights DXT data?
  - Identify I/O bottlenecks
  - Hint which optimizations we should apply
The DXT Explorer Tool

- Darshan can collect fine grain traces with DXT
  - No tool to visualize and explore yet
  - Static plots have limitations

By visualizing the application behavior, we are one step closer to optimize the application.

There is still a lack of translation from I/O bottlenecks to optimizations.
The DXT Explorer Tool

- Darshan can collect fine grain traces with DXT
  - No tool to visualize and explore yet
  - Static plots have limitations

- Features we seek:
  - Observe POSIX and MPI-IO together
  - Zoom-in/zoom-out in time and subset of ranks
  - Contextual information about I/O calls
  - Focus on operation, size, or spatiality

- By visualizing the application behavior, we are one step closer to optimize the application
- There is still a lack of translation from I/O bottlenecks to optimizations
Exploring I/O Traces with DXT Explorer | BERKELEY LAB

github.com/hpc-io/dxt-explorer

docker pull hpcio/dxt-explorer
collective calls translate into several POSIX calls

same amount of data in each timestep

stragglers observed in different ranks

some stragglers make the collective calls take longer

OST information will show up if available (Lustre)

Rank: 25
Operation: write
Duration: 12.07 seconds
Size: 32768 KB
Offset: 16273899520
small operations (HDF5 metadata) in some ranks

stragglers observed in collective operations

different controls available to interact and explore
Rank: 128
Operation: write
Duration: 0.095 seconds
Size: 3247.67 KB

contextual information available for each operation
Conclusion

- **DXT Explorer**
  - Adds an interactive component to *Darshan DXT* trace analysis
  - Moves a step closer towards connecting the dots between bottleneck detection and tuning

- There is still the need for further R&D
  - How can we better report findings to end-users?
  - How can we automatically map performance problems to tuning options?
  - How can we provide recommendations?

```
docker pull hpcio/dxt-explorer

github.com/hpc-io/dxt-explorer
```
Visualizing Darshan Traces

Jean Luca Bez
Lawrence Berkeley National Laboratory

Jean Luca Bez | jlbez@lbl.gov
Suren Byna | sbyna@lbl.gov

SC’21 BoF: Analyzing Parallel I/O