IOPathTune: Adaptive Online Parameter Tuning for Parallel File System I/O Path

Md. Hasanur Rashid, Dong Dai (Advisor)
Department of Computer Science, University of North Carolina at Charlotte

1. Motivations

- Parallel file systems like Lustre contain complicated I/O paths from clients to storage servers. Its efficiency is critical for performance.
- I/O path requires proper settings of multiple parameters. The default settings often fail to deliver optimal performance, especially for diverse workloads in the HPC environment.
- Existing tuning strategies are limited in terms of being *adaptive*, *timely*, and *flexible*.
- We propose *IOPathTune*, which adaptively tunes PFS I/O Path online from the *client side*.

2. IOPathTune Goals

- ***Adaptive.***
  - Work adaptively when workloads change.
  - Response to runtime such as I/O contentions.
- ***Online.***
  - Adapt quickly when system status changes.
  - Change parameters without remounting.
- ***Flexible.***
  - Tune parameters for multiple clients separately based on their own executions.
  - Avoid relying on expensive probing, communication, or profiling.

3. IOPathTune Design and Implementation

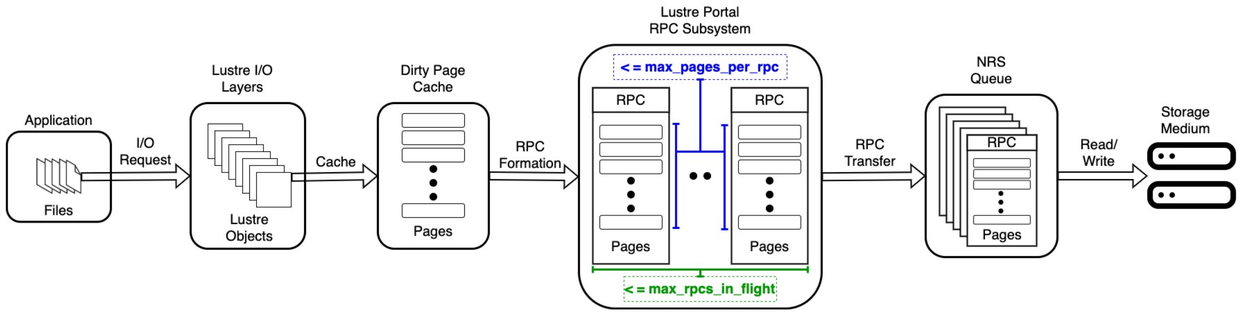

Figure 1: Lustre I/O Path

- ***What do we observe to tune?***
  - IOPathTune does not probe storage servers or other compute nodes. It solely depends on statistics collected by the PFS client library.
- ***What do we tune for PFS I/O Path?***
  - IOPathTune tunes two client-side OSC-level parameters: `max_pages_per_rpc` and `max_rpcs_in_flight` (see Figure 1).
- ***How often does IOPathTune work?***
  - Tune every ten seconds and collects stat information snapshot in the middle (see Figure 2).
- ***How do we tune?***
  - Similar to TCP congestion control, the tuning action is either to continue or reverse previous actions (see Figure 3).
  - We multiply or to divide the parameter value by two each time.

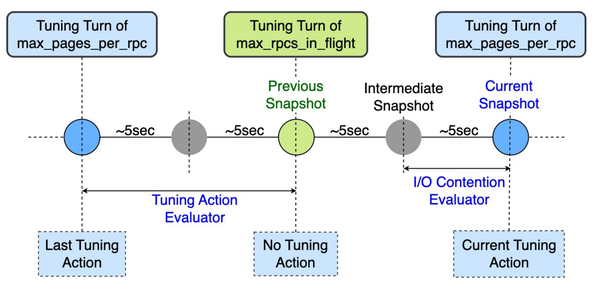

Figure 2: Snapshot Collection

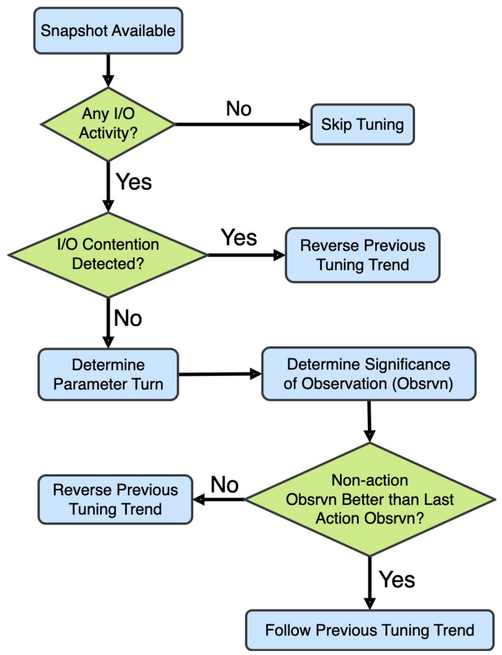

Figure 3: Heuristic Algorithm Flowchart

4. Evaluations

- ***Setup.***
  - [Experiment Platform] CloudLab [1] c220g5 machines: 1 MDS, 4 OSS, and five compute nodes.
  - [Software] Lustre 2.12.5 file system.
  - [Workloads] 20 different Filebench [2] workloads (see Table 1).
  - [Execution Environment] Both single and multiple client(s).
- ***Brief Results.***
  - IOPathTune either improves or performs on par with slight degradation over the default configurations in all standalone single-client workload executions (see Table 1).
  - IOPathTune gains improvements as high as **231%, 113%, 96%** for fivestreamwriternd, sequential, and whole-file read-write standalone workload executions (see Table 1).
  - IOPathTune adapts to the near-optimal parameter configurations quickly upon workload changes (see Figure 4).
  - IOPathTune maintains appropriate parameter configurations for multiple clients executing different workloads (see Table 2).
  - IOPathTune, in comparison with default configuration achieves **129% improvement** and in comparison with CAPES [3] execution achieves **89% improvement** on the overall bandwidth of the cluster (see Table 2).

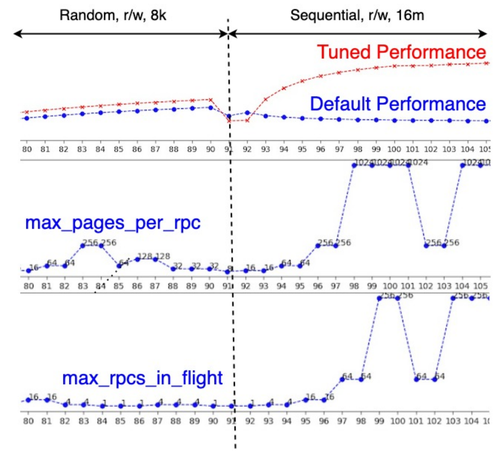

Figure 4: Dynamic Workload Change Execution

Table 1: Single Client Standalone Workload Executions

Workload Name	I/O Request = 8KB		I/O Request = 1MB		I/O Request = 16MB	
	I/O BW Change(%)	*Tuned Configurations*	*I/O BW Change(%)*	*Tuned Configurations*	*I/O BW Change(%)*	*Tuned Configurations*
Random Write	7.82	(32, 32)	**22.97**	(256, 4)	10.93	(1024, 4)
Fivestream Random Write	**64.46**	(32, 16)	**231.98**	(8, 1)	**43.44**	(8, 16)
Random Read-Write	-7.46	(8, 4)	5.57	(256, 1)	-2.91	(128, 16)
Sequential Write	-4.39	(256, 4)	-0.73	(256, 1)	7.56	(1024, 16)
Fivestream Sequential Write	-7.29	(256, 64)	3.75	(512, 128)	-7.59	(1024, 64)
Sequential Read-Write	4.03	(1024, 64)	**113.19**	(1024, 16)	**72.6**	(1024, 16)
Whole File Write					**86.45**	(32, 8)
Whole File Read-Write					**96.58**	(8, 8)

Table 2: Multiple Client Different Workload Executions

Workload Name	Client Name	Default Execution		CAPES Execution		IOPathTune Execution	
		I/O BW(MB/s)	*Default Configurations*	*I/O BW(MB/s)*	*Tuned Configurations*	*I/O BW(MB/s)*	*Tuned Configurations*
Fivestream Random Write	node1	385.4	(1024, 8)	237	(200, 108)	**2627.9**	(32, 32)
Random Write	node2	95.2	(1024, 8)	101.4	(200, 108)	**206.3**	(128, 8)
Random Read-Write	node3	2127.6	(1024, 8)	**4209.3**	(200, 108)	3199.8	(512, 16)
Sequential Read-Write	node4	639.2	(1024, 8)	630.8	(200, 108)	**1134.6**	(512, 128)
Whole File Read-Write	node5	1682.3	(1024, 8)	784.3	(200, 108)	**4135**	(8, 2)
Total Multi-client BW (MB/s)		4929.7		5962.8		**11303.6**	

5. Conclusions

Our study has shown how the proposed algorithm can perform adaptive online parameter tuning without characterizing workloads, doing expensive profiling, and performing expensive communications. We hope this approach will welcome more attention towards researching inexpensive yet effective solutions in system research.

6. Future Research

We like to scale our algorithm to accommodate tuning more parameters following this heuristic approach. We would also like to test it out in real-world HPC facilities to observe how much improvement the solution brings regarding I/O performance.

7. References

[1] Duplyakin, Dmitry, et al. "The Design and Operation of CloudLab." *2019 USENIX annual technical conference (USENIX ATC 19)*. 2019.

[2] Tarasov, Vasily, Erez Zadok, and Spencer Shepler. "Filebench: A flexible framework for file system benchmarking." *USENIX; login* 41.1 (2016): 6-12.

[3] Li, Yan, et al. "CAPES: Unsupervised storage performance tuning using neural network-based deep reinforcement learning." *Proceedings of the international conference for high performance computing, networking, storage and analysis*. 2017.