

Active Flash: Out-of-core Data Analytics on Flash Storage

Simona Boboila¹, Youngjae Kim², Sudharshan S. Vazhkudai², Peter Desnoyers¹, Galen M. Shipman²

¹Northeastern University, ²Oak Ridge National Laboratory

¹{simona, pjd}@ccs.neu.edu, ²{kimy1, vazhkudaiss, gshipman}@ornl.gov

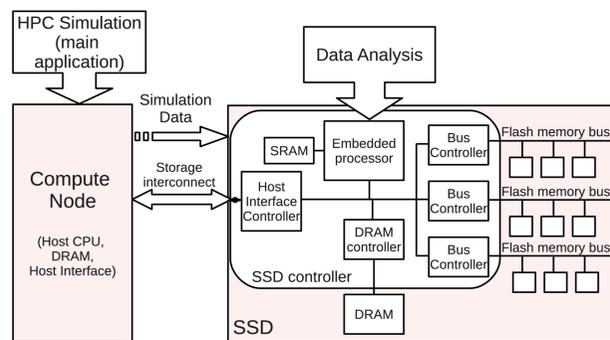
Motivation

Large scale multi-core systems:

- I/O bandwidth has failed to keep pace with computation
 - ➔ reduce redundant I/Os between storage and host CPU;
- power consumption constraints (megawatts)

Architecture

Active Flash: Data analysis is moved from the host CPU to the storage controller, closer to where data already resides.



Feasibility of Active Flash:

- High I/O bandwidth in SSDs: 100–500 MB/s.
- Availability of idle times in workloads due to: low I/O latencies, HPC I/Os burstiness.
- High-performance embedded processors: 80-800 MHz SSD controllers; 1-2 GHz “mobile class” processors; 1-4 cores.

Evaluation on realistic data analysis applications

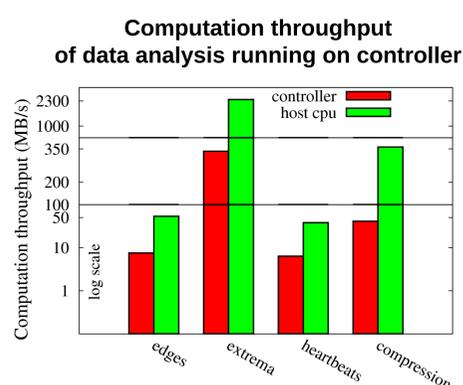
Applications:

- Data compression on scientific data
- Edge detection on atmospheric data
- Local extrema detection on medical data
- Heartbeat detection on medical data

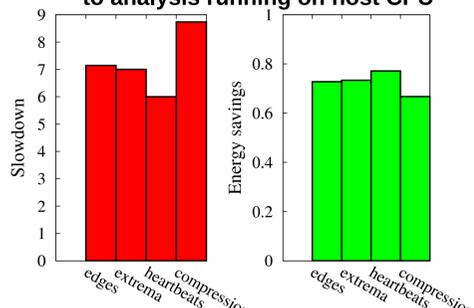
Test environment:

- **Storage controller:** dual-core 1 GHz ARM Cortex-A9 on the Pandaboard system
- **Host CPU:** Intel Core 2 Quad CPU Q6600 2.4 GHz
- Measured computation phase (no I/O) on one processor core with no competing processes.

Performance-Energy trade-offs:



Slowdown and energy savings of data analysis. Compares analysis running on controller to analysis running on host CPU



$$\text{Slowdown: } S = \frac{t_{ssd}}{t_{host}}$$

Energy savings:

$$\Delta P = P_{load} - P_{idle}$$

$$\Delta E = 1 - \frac{\Delta E_{ssd}}{\Delta E_{host}}$$

$$= 1 - \frac{t_{ssd}}{t_{host}} \frac{\Delta P_{ssd}}{\Delta P_{host}}$$

Scheduling data analysis on flash

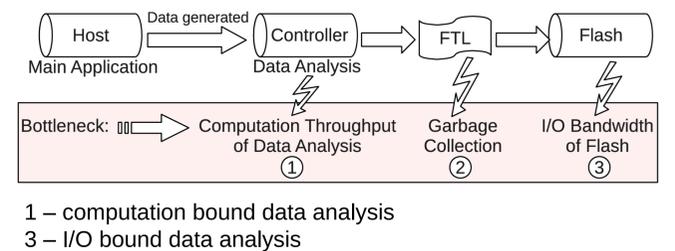
- **On-the-fly data analysis** – while data is still in controller's DRAM
 - ➔ reduce I/O traffic inside SSD
 - ➔ data generation rate \leq analysis throughput (DRAM size limitation)
- **Idle-time data analysis** – data written to SSD, analyzed later during idle times
 - ➔ most HPC I/O workloads are bursty, thus \exists idle times
 - ➔ compared to on-the-fly:
 - higher I/O traffic inside the SSD, but
 - higher data generation rates sustained (limited only by SSD write rate)
- **Idle-time data analysis with GC management**
 - ➔ when no data to process, schedule GC during idle times
 - ➔ complements the existing GC policy (i.e. invoked when free blocks $\# <$ threshold)

Simulator implementation:

- Implemented the scheduling policies in the Microsoft Research SSD simulator (event-driven).
- Additional parameters:

Parameter	Value
Computation time (per page of input)	application-specific
Data reduction ratio	application-specific
GC-idle threshold (fraction of reserved space)	0.9

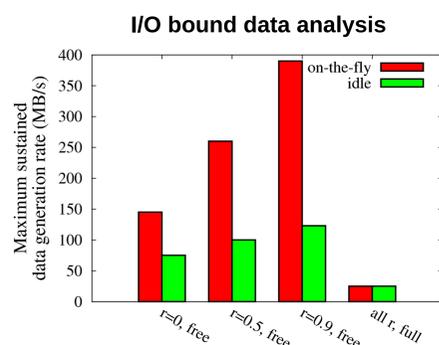
Bottlenecks:



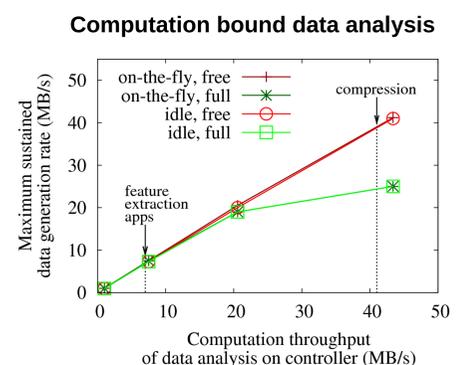
1 – computation bound data analysis
3 – I/O bound data analysis

Scheduling results:

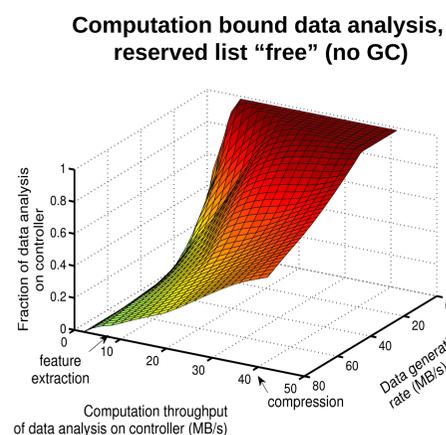
- What is the maximum data generation rate that controller-resident data analysis can sustain, i.e. controller finishes the entire analysis in the time frame while the host CPU is busy generating data?



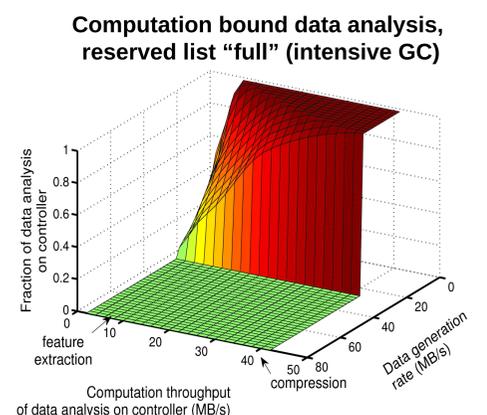
- ➔ **I/O bound data analysis:** SSD I/O bandwidth (145 MB/s) < computation throughput (390 MB/s)
- ➔ **Computation bound data analysis:** SSD I/O bandwidth (145 MB/s) > computation throughput ([1 .. 43] MB/s)
- r = data reduction (output size / input size)
- free = analysis starts with reserved list “free” (no GC)
- full = analysis starts with reserved list “full” (intensive GC)



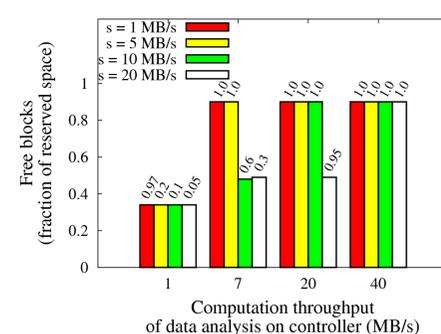
- At higher data generation rates, what fraction of data analysis can be accommodated on the controller in the time frame while the host CPU is busy generating data?



- ➔ Idle-time scheduling (both left and right).
- ➔ Impact of GC: GC saturates the SSD (right) at data generation rates > 25 MB/s.
- ➔ Hybrid Active Flash model: controller+host CPU for data analysis.



- What fraction of the reserved list is GC able to clean (analysis starts with reserved list “full”)?
 - ➔ GC can clean at most (fraction of reserved list):
 - GC-hard-threshold: up to 0.33 (hard threshold = 0.33);
 - GC-idle-time: from 0.33 to 0.9 (pro-active cleaning).



s = data generation rate
bar labels = fraction of data analysis finished on the controller during the data generation time