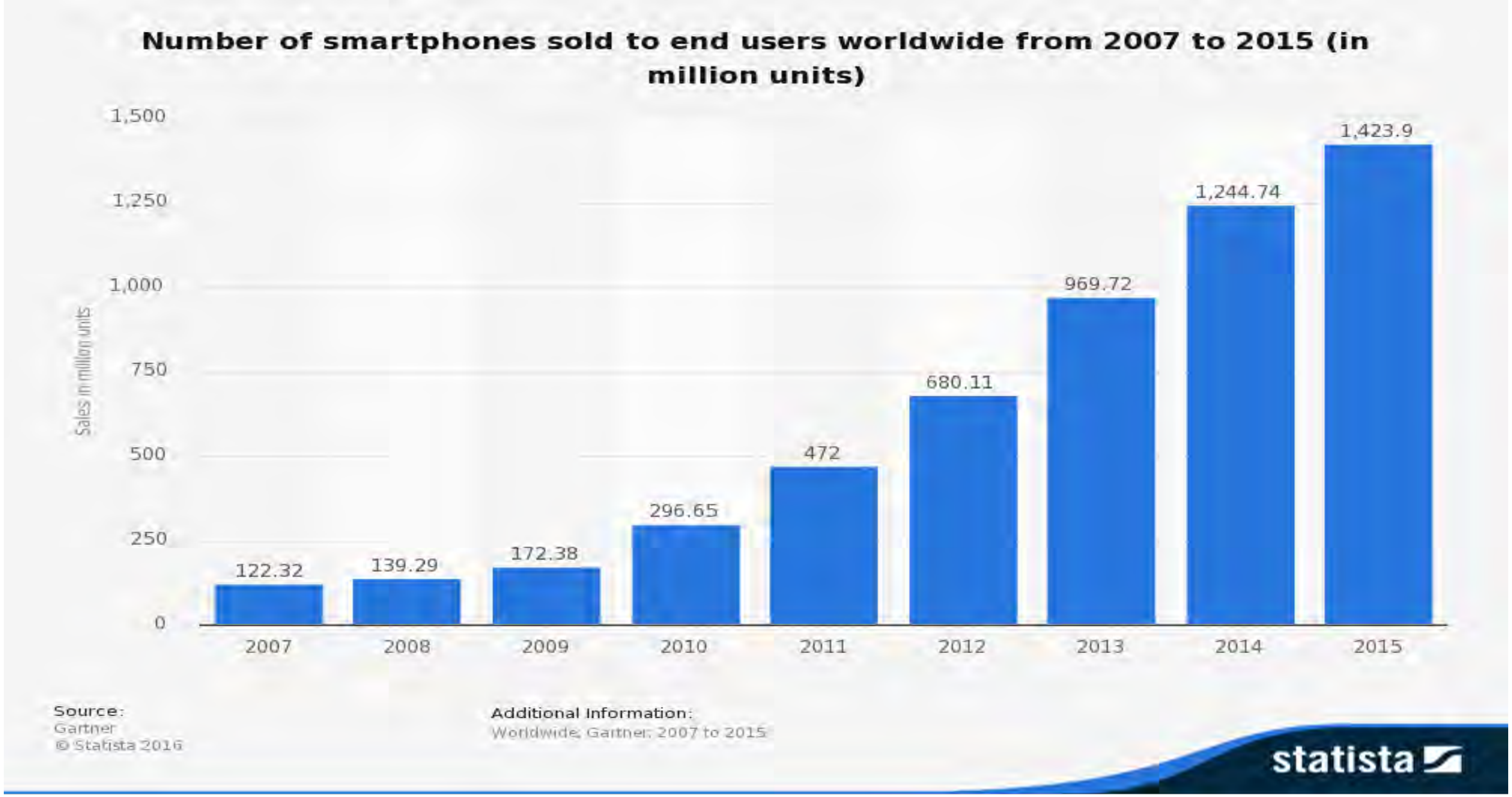
Understanding Storage I/O Behaviors of Mobile Applications

Jace Courville
jcourv@csc.lsu.edu

Feng Chen
fchen@csc.lsu.edu

Louisiana State University
Department of Computer Science and Engineering

The Rise of the Smartphone



http://www.statista.com/statistics/263437/global-smartphone-sales-to-end-users-since-2007/

- Smart device use has steadily increased since 2007
- Users are switching to these devices for daily computing tasks

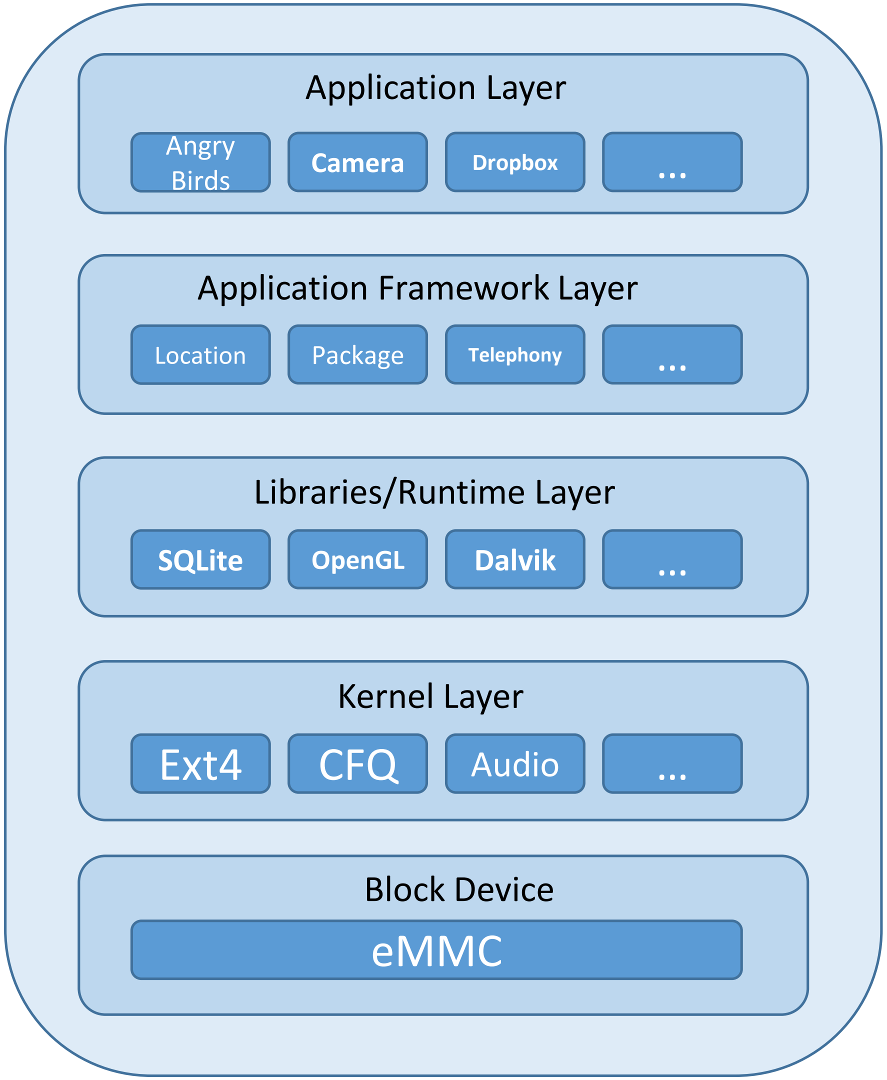
Unique Behaviors of Mobile Applications

- Flash-based storage medium
  - High read performance, poor random write performance
- Latencies have a greater impact on device usability
  - Optimizations need to be latency-oriented
- Distinct software stack and distinct app characteristics

android

iPhone

The Android Architecture



- Applications are considered "*users*" with their own unique ID and set of permissions
- Applications run in a protected environment and privileged operations are encapsulated in a small set of API interfaces
- Libraries such as SQLite are heavily used in **nearly all** mobile apps

Prior wisdom may not apply

Key Questions

- How much do storage I/Os impact workload performance?
- Which type of storage I/Os contribute the most to latency?
- Are there any consistent trends in application performance?
  - Are behaviors different over different categories of workloads?
- What are the systems implications of storage I/O Latency?

Experimental Setup

- Google Nexus 5, 32 GB eMMC storage, 2 GB RAM
- AOSP Android 5.1 OS / Linux kernel 3.4.0
- blktrace / blkparse used to collect and interpret I/Os
  - Traces are stored on ramfs to eliminate blktrace overhead
  - Device restarted between each test to remove variance
  - blktrace started following end of interaction
- Metrics Gathered:
  - I/O Request Size, I/O Latency
  - Information Between Successive Flushes
  - Locality
  - Percentage of I/O time

Workloads

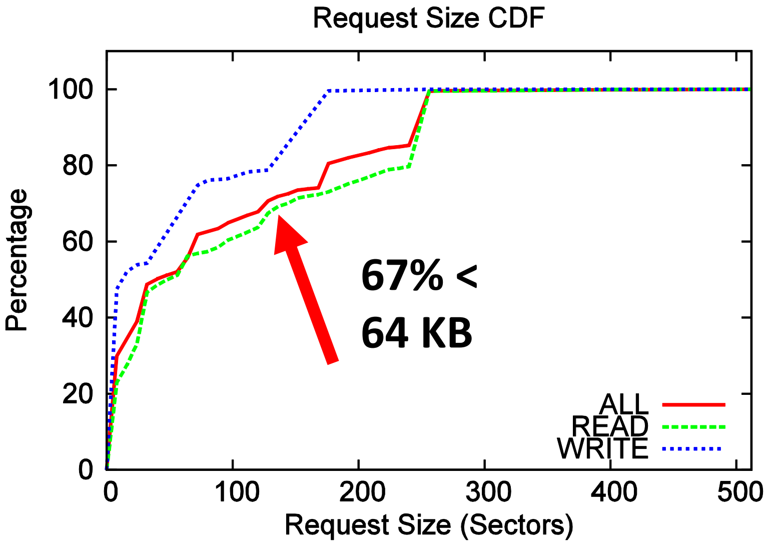
- 13 Workloads from 5 categories representing real-world scenarios

Workload Name	Workload Type	R/W Ratio	Read-based	Write-based	Description
Angry Birds	Game	**2.03**/1	**X**		Load the Angry Birds Application
App Removal	Device Utility	**1.35**/1	**X**		Uninstall an Application
Batch Uninstall	Device Utility	1/**2.79**		**X**	Uninstall several Applications through ADB at once
Camera	Multimedia	1/**9.12**		**X**	Default Camera used to take 3 pictures in sequence
Burst Mode Camera	Multimedia	1/**204.1**		**X**	Burst Mode Camera app used to take 100 photos in burst
Video Recording	Multimedia	1/**4.25**		**X**	Uses default Camera to record a 5 second video
Video Playback	Multimedia	**1.81**/1	**X**		Plays back the recorded 5 second video
Add Contact	Productivity	1/**2.07**		**X**	New contact is added through the Contacts app
Sync Dropbox	Network	1/**5.63**		**X**	Links an existing DropBox account to the device and syncs
Sync E-Mail	Network	1/**4.25**		**X**	Links an existing E-mail account to the device and syncs
Web Request	Network	1/**1.47**		**X**	Load the Facebook web site through the default browser
Route Plot	Network	1/**2.54**		**X**	Plots a GPS route using the Google Maps app
MP3 Stream	Network	1/**41.8**		**X**	Streams 15 seconds of a song in the Spotify app

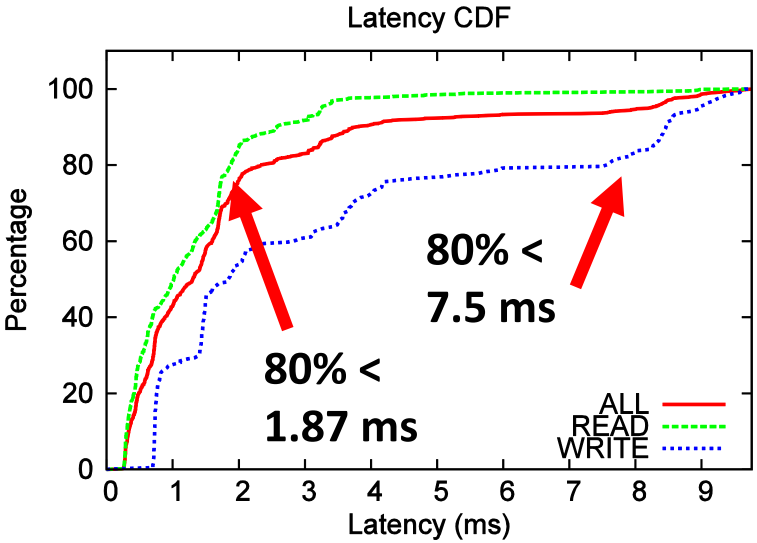
Outline of Experiments

- Basic Observations
  - Two key factors: Request Size and Latency
- Flushing Behavior
  - Directly impacts I/O speed on NAND flash-based storage
  - Requests, Total Size, Time – Between Successive Flushes
- Access Locality
  - Has strong implications to cache efficiencies
- Total Storage I/O Latency impact
  - What percentage of runtime is storage I/O latency?

Basic Observations: Angry Birds



67% <
64 KB

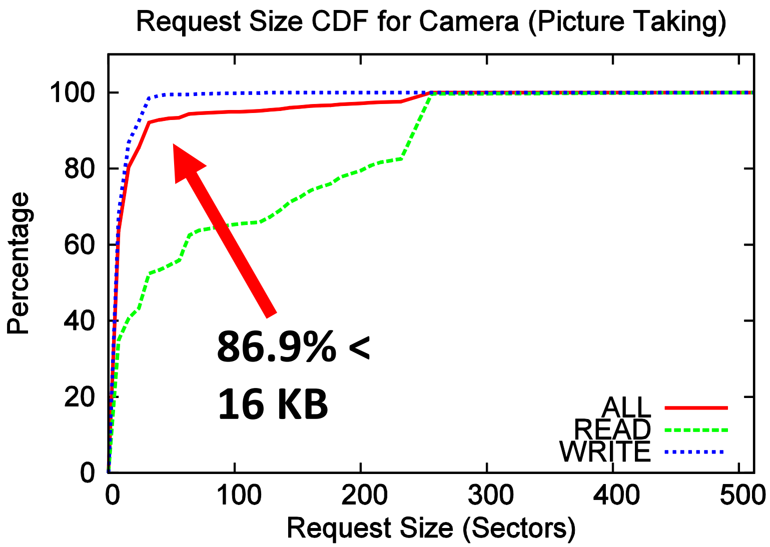


80% <
1.87 ms

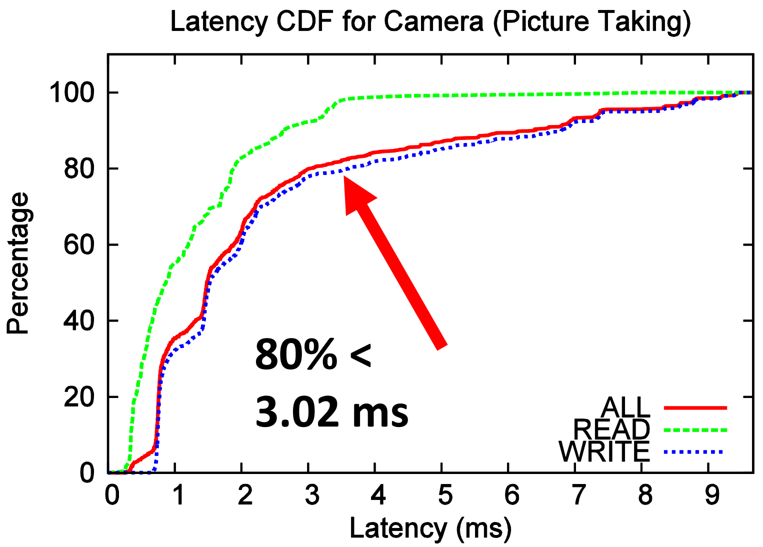
80% <
7.5 ms

- Average case – Small request sizes of varying latency
- Read-Heavy Workload
  - Highest number of reads of any workload (567)
- 67.8% of all I/Os are smaller than 64 KB
- Writes longer than reads

Basic Observations: Camera – Normal Mode



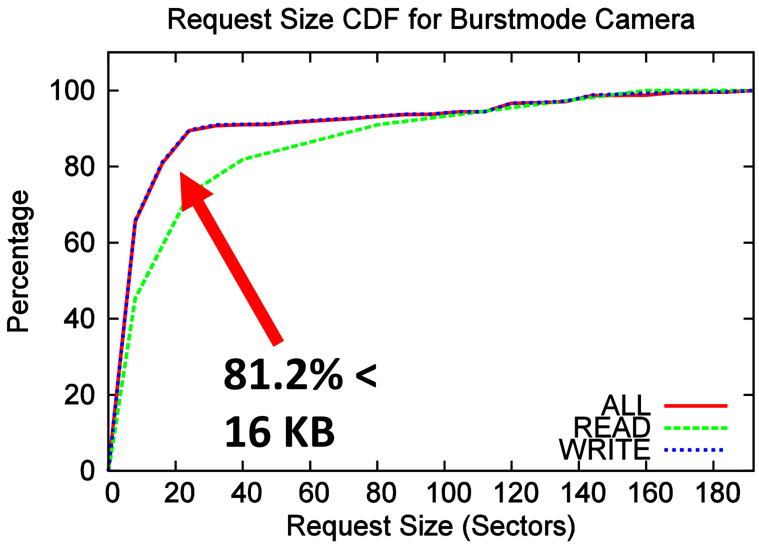
86.9% <
16 KB



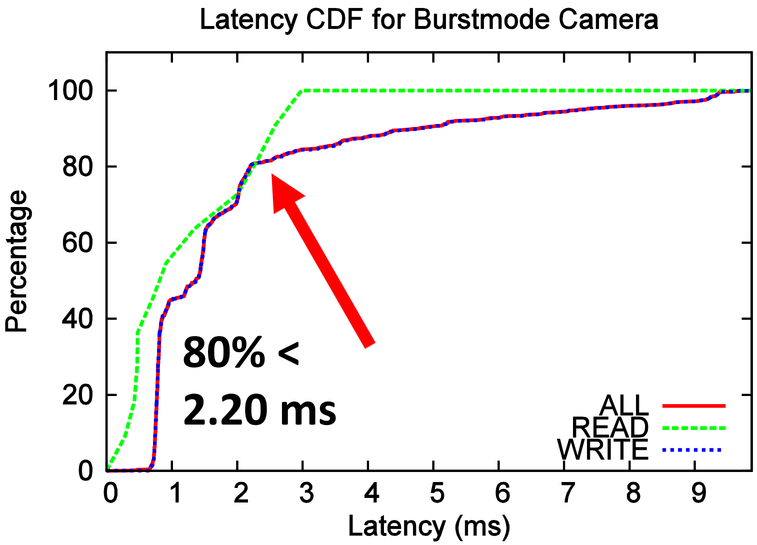
80% <
3.02 ms

- Highly write-heavy – 9.12 writes to 1 read (3rd highest)
    - 2nd highest total writes (2090)
- All writes are very small – 86.9% smaller than 16 KB

Basic Observations: Camera – Burst Mode



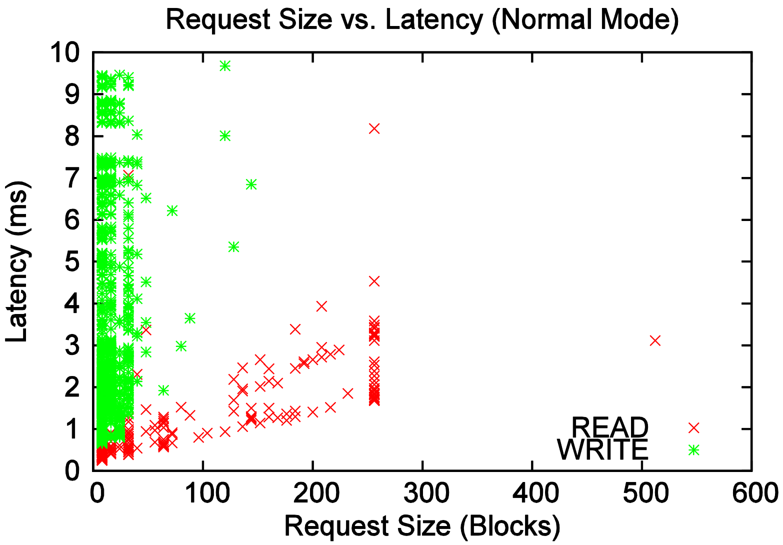
81.2% <
16 KB

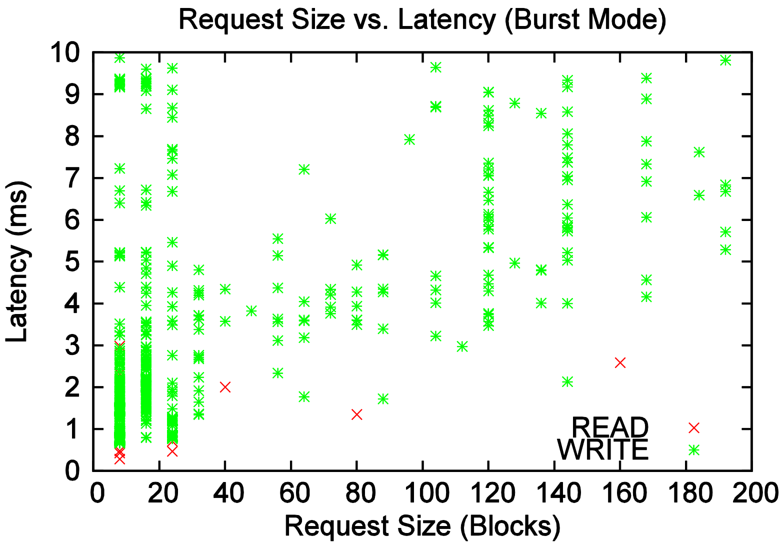


80% <
2.20 ms

- Most write-heavy workload (204.1 writes to every 1 read)
  - Most writes of any workload at 2246
  - Fewest reads of any workload at 11
  - Writes are more variable in size
- Only 156 more reads than the Normal Mode workload

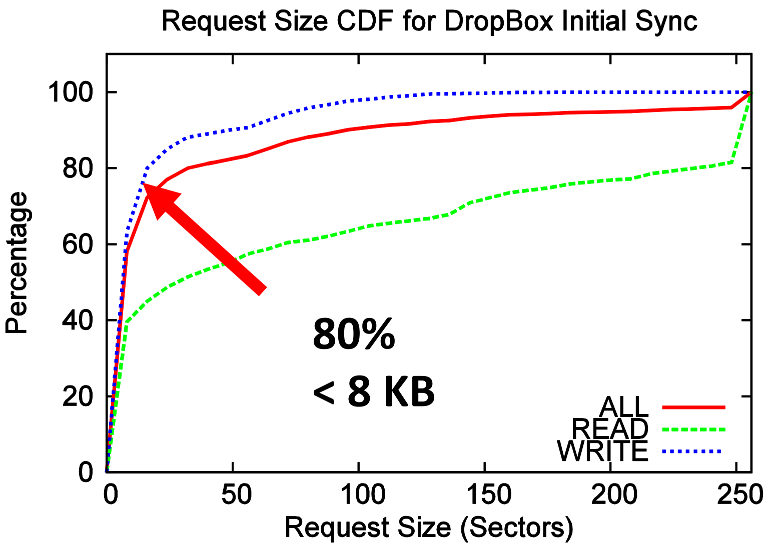
Basic Observations: Camera

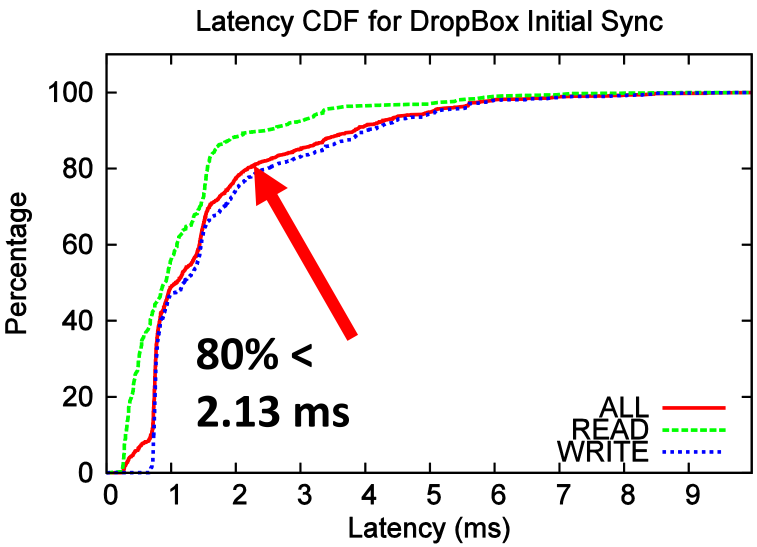




- Both Camera modes experience variable latency for I/O writes
- Normal mode workload sees smaller writes, reads
- Burst workload sees very few reads, much larger writes

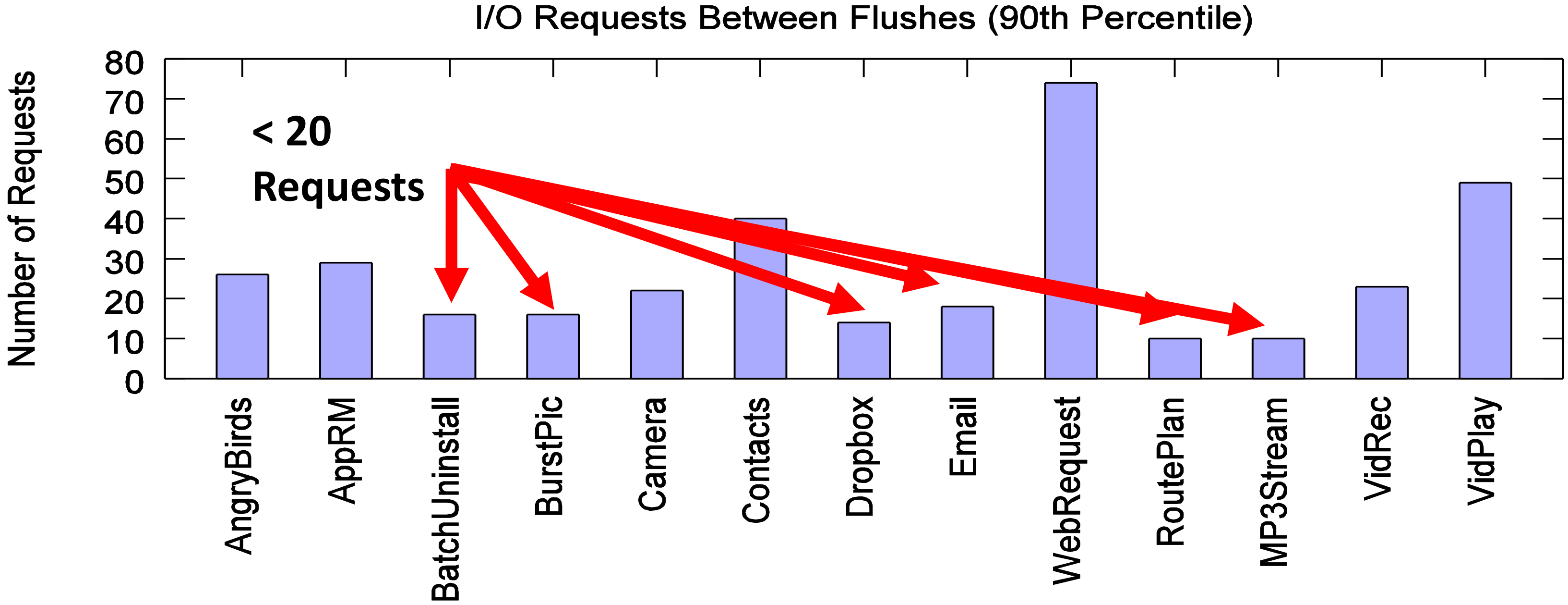
Basic Observations: Dropbox Sync





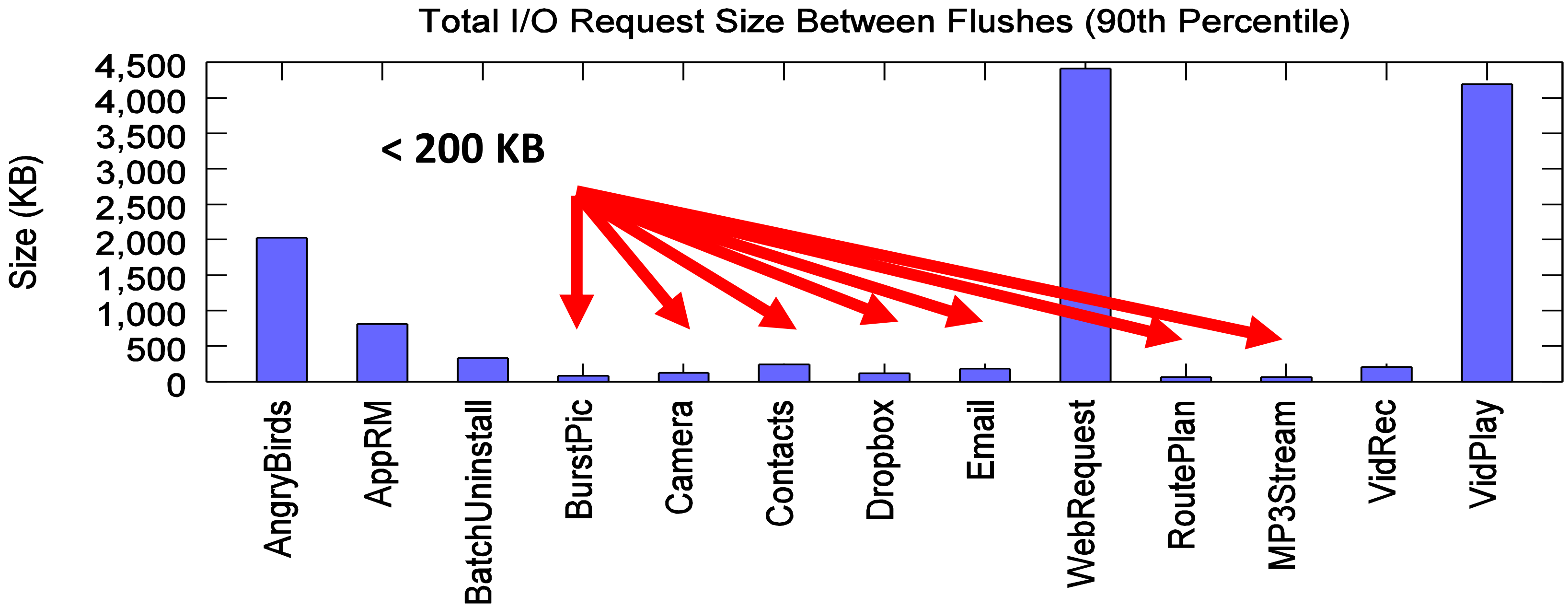
- Network-based workload – Majority small writes (80% < 8 KB)
- Compared to other workloads, reads are larger
- All writes have highly variable latencies

Flushing Behavior



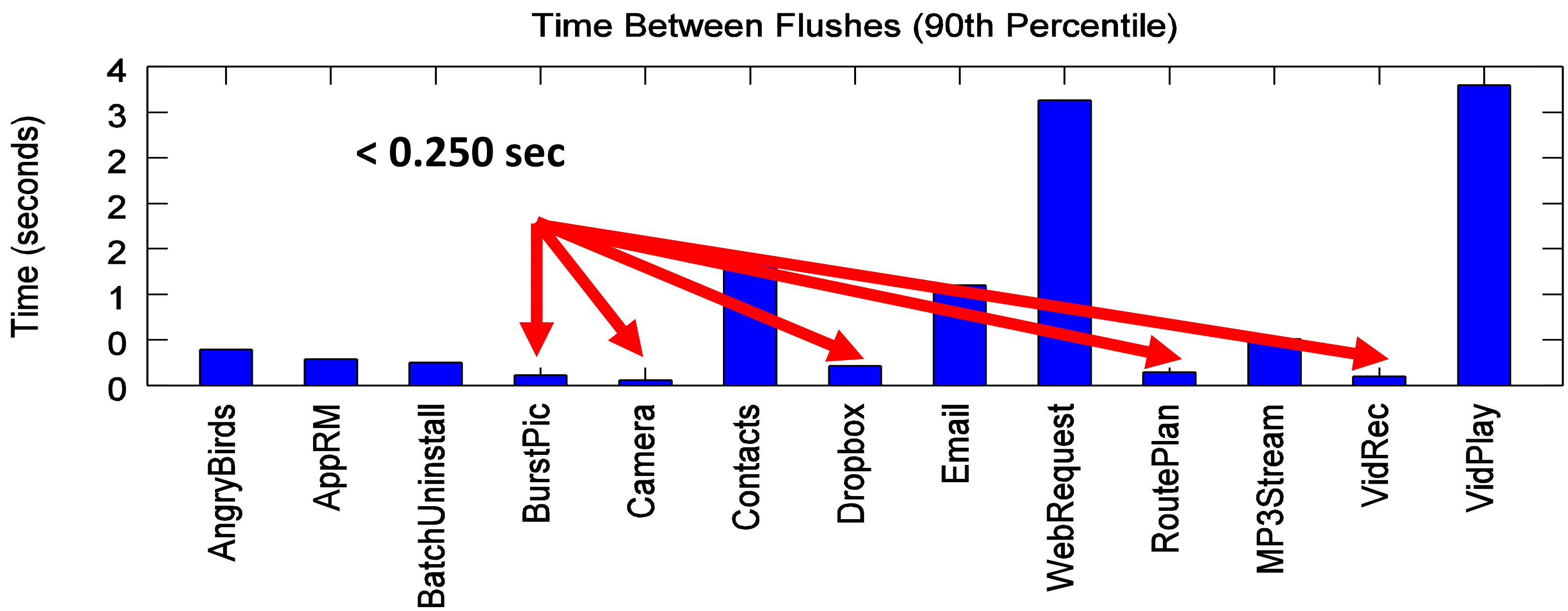
- Application Developers may wish to ensure data persistence
- Android OS uses flush operation to send buffered data to storage
- Too much flushing can be a bad thing
  - Can result in increased latency, therefore decreased performance
- Trend of excessive flushing is common

Flushing Behavior



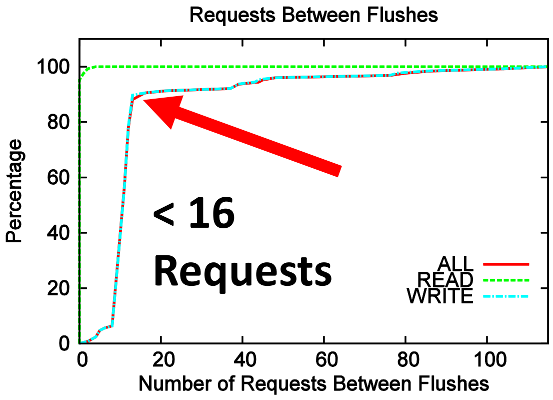
- Application Developers may wish to ensure data persistence
- Android OS uses flush operation to send buffered data to storage
- Too much flushing can be a bad thing
  - Can result in increased latency, therefore decreased performance
- Trend of excessive flushing is common

Flushing Behavior

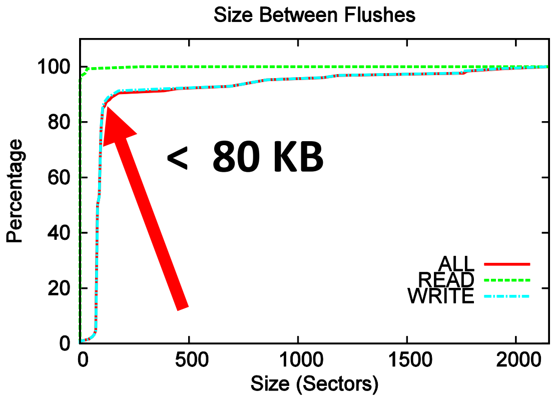


- Application Developers may wish to ensure data persistence
- Android OS uses flush operation to send buffered data to storage
- Too much flushing can be a bad thing
  - Can result in increased latency, therefore decreased performance
- Trend of excessive flushing is common

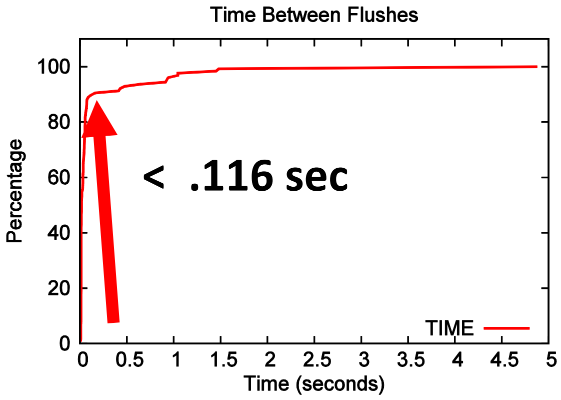
Burst Mode Camera


Requests Between Flushes

< 16 Requests


Size Between Flushes

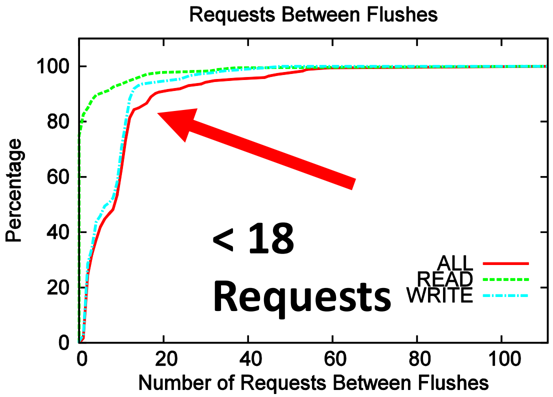
< 80 KB


Time Between Flushes

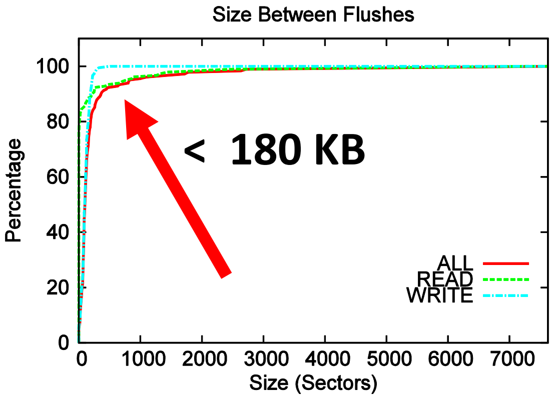
< .116 sec

- 90% of Flushes have < 16 I/O requests between successive flush operations.
  - < 80 KB of Data and < .116 sec between flushes
- Very aggressive flushing – Extremely short iterations between flushes

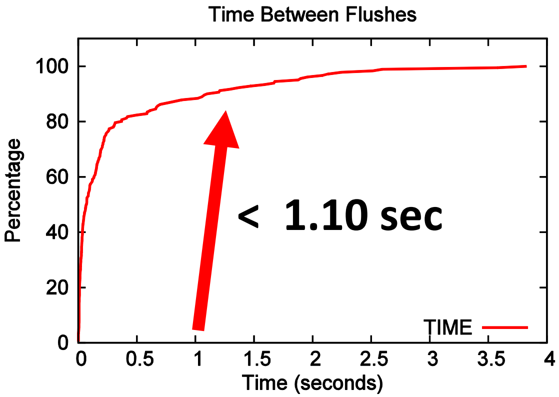
E-mail Sync


Requests Between Flushes

< 18 Requests


Size Between Flushes

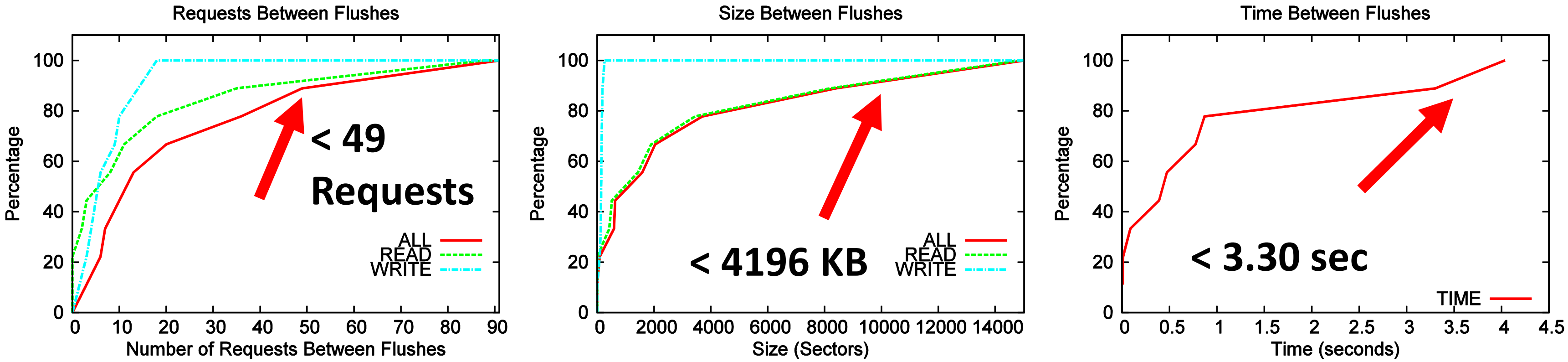
< 180 KB


Time Between Flushes

< 1.10 sec

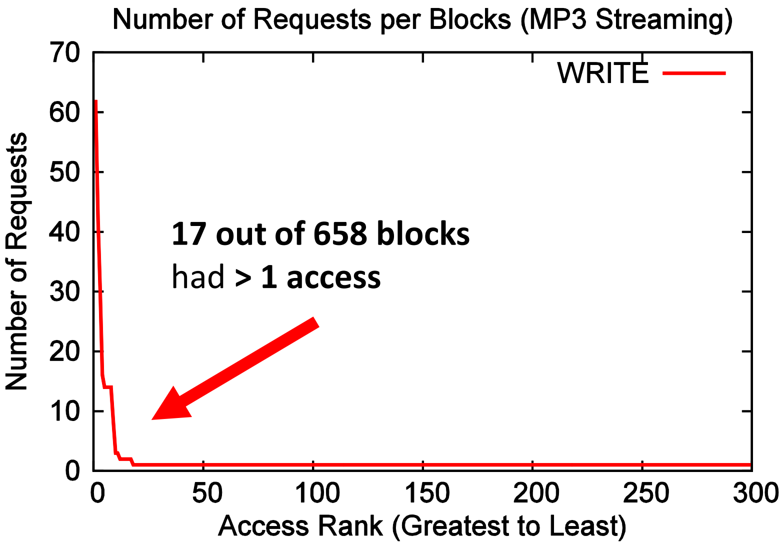
- 90% of Flushes have < 18 I/O requests between successive flush operations.
  - < 180 KB of Data and < 1.10 sec between flushes
- Data persistence is desired, so we see utilization of flush operations

Video Playback

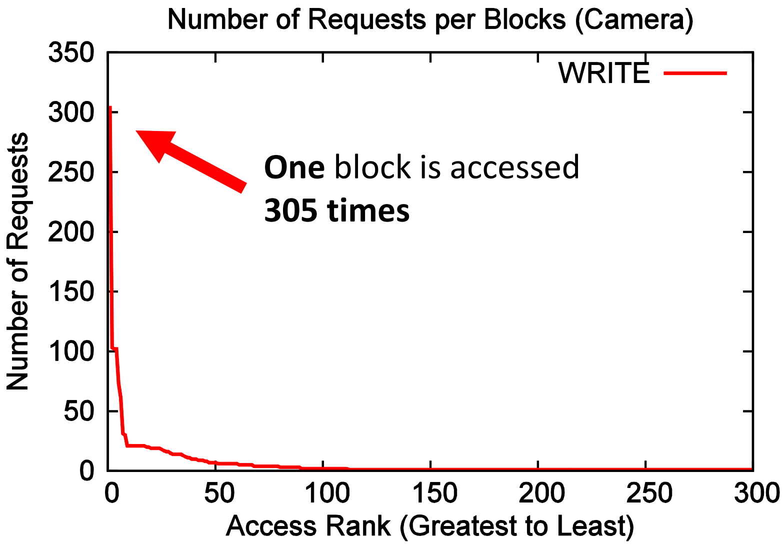


- 90% of Flushes have < 49 I/O requests between successive flush operations.
  - < 4196 KB of Data and < 3.30 sec between flushes
- I/O writes not heavily used -- not as important to make any data persistent

Locality



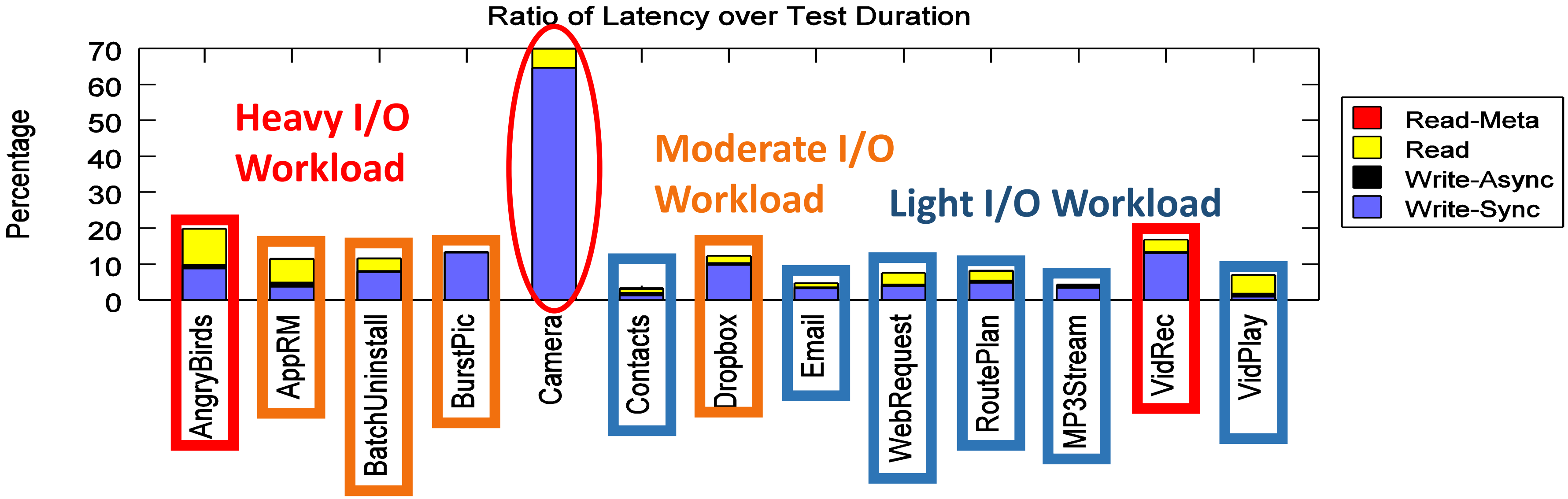
17 out of 658 blocks had **> 1 access**



One block is accessed **305 times**

- A common trend – Very few blocks experience multiple accesses
- Camera workload had one block re-accessed 305 times
- Only top 300 most accessed blocks shown
  - MP3 Streaming has 658 accessed block – Camera has 3293
- Nearly all workloads saw reads as single access only

Impact of Storage I/O latency



- The impact of Storage I/O latency varies by workload
- Camera is the most affected, at nearly 70%
- Asynchronous Writes and Reads were the direct contributors
  - Metadata Reads and Asynchronous writes had little to no impact
- Storage I/O Latency impact may not be user-perceivable

System Implications

- I/O Writes are small with varying latency
  - Small writes range from 1 ms to 10 ms of latency
  - Category independent trend – Dropbox was 5th most affected workload
- Aggressive flushing is very common
  - Data safety is a concern for developers – results in aggressive flushing
  - Resulting small writes will magnify slow write performance of flash storage
- I/O Reads happen only once in nearly all workloads
  - Confirmed by reducing available RAM to 1 GB
  - Sufficient RAM availability has the most impact
- Synchronous writes are the most common – and the biggest issue
  - By numbers, Synchronous Writes and Reads were similar
  - Metadata Reads / Asynchronous writes uncommon with minimal impact
- Storage I/O impact varies by workload
  - Camera workload much larger – next most impacted was 20%
  - May not have as much as a user perceivable impact as previously thought

Conclusions

- There is a definite space for storage I/O optimization
- Small, synchronous writes are the biggest cause for I/O latency
- Reducing flushing will negate much of the latency caused by I/Os
- Impact of I/O latency is application and workload dependent
- Any solution must be customized to the individual workload

Thank You!

Jace Courville
jcourv@csc.lsu.edu

Feng Chen
fchen@csc.lsu.edu