Optimizing Disk Layouts for Mixed Sequential Read Workloads

Rickard Faith and Stephen Daniel
NetApp

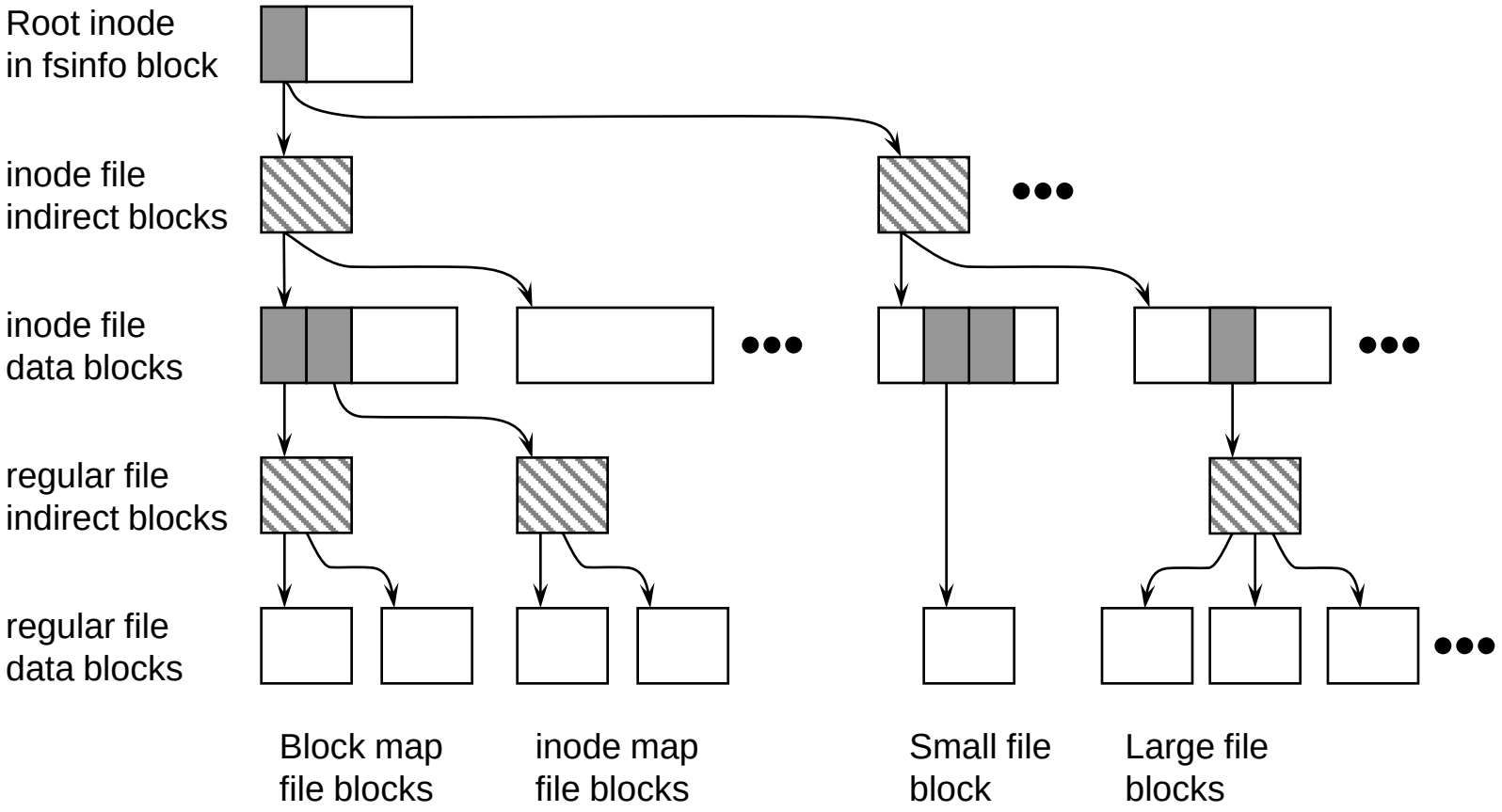
Outline of Talk

- Introduction
- Problem Statement
- Optimization Technologies
- Analysis and Results
- Future Directions and Conclusion

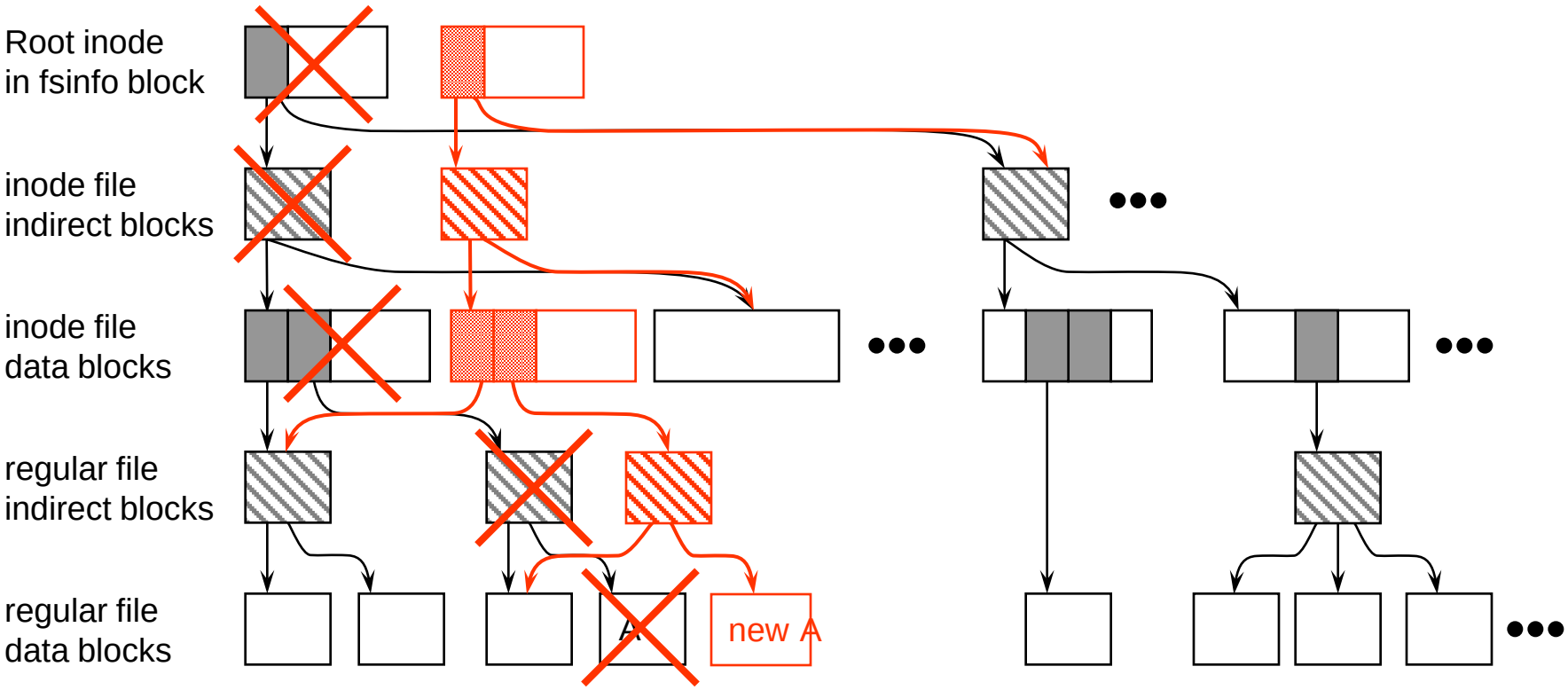
Introduction

- What is a shadow paging file system?
- How random writes are optimized
- The sequential read after random write problem

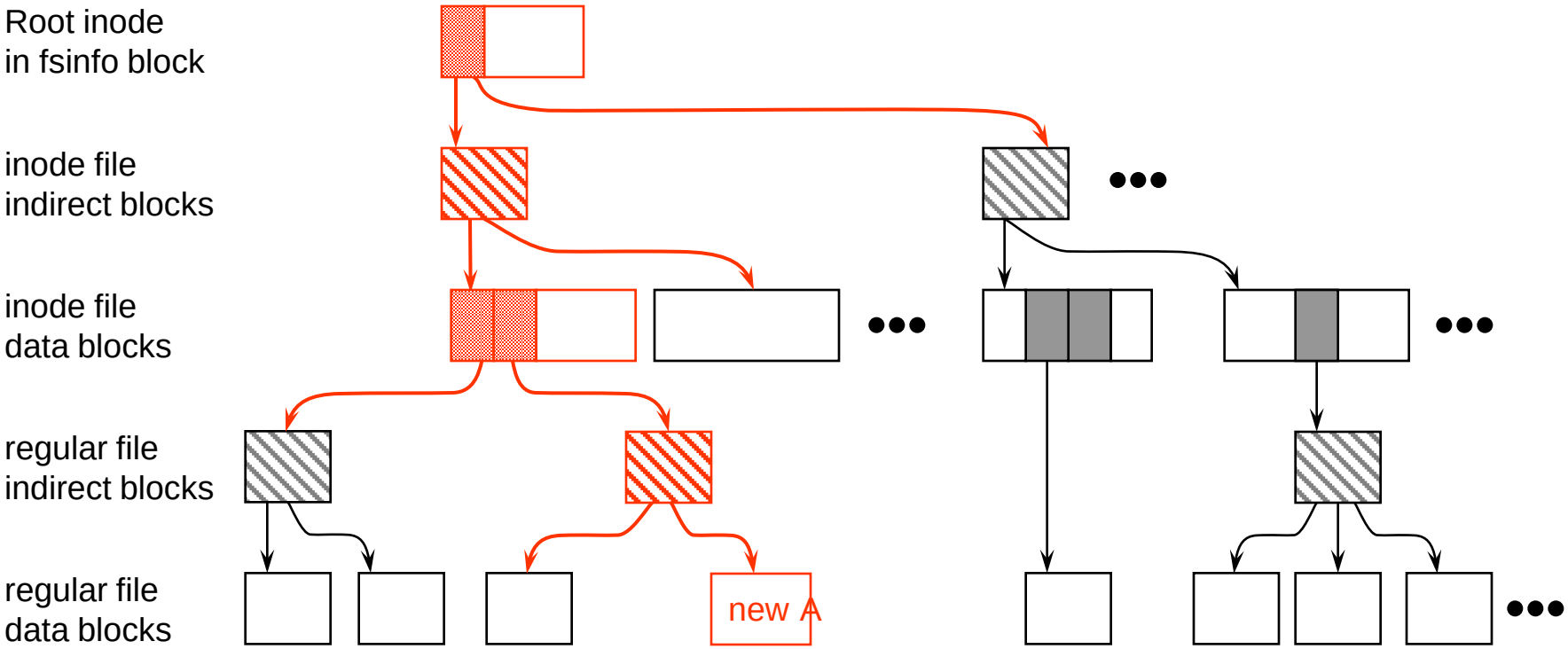
Shadow Paging File System Tree



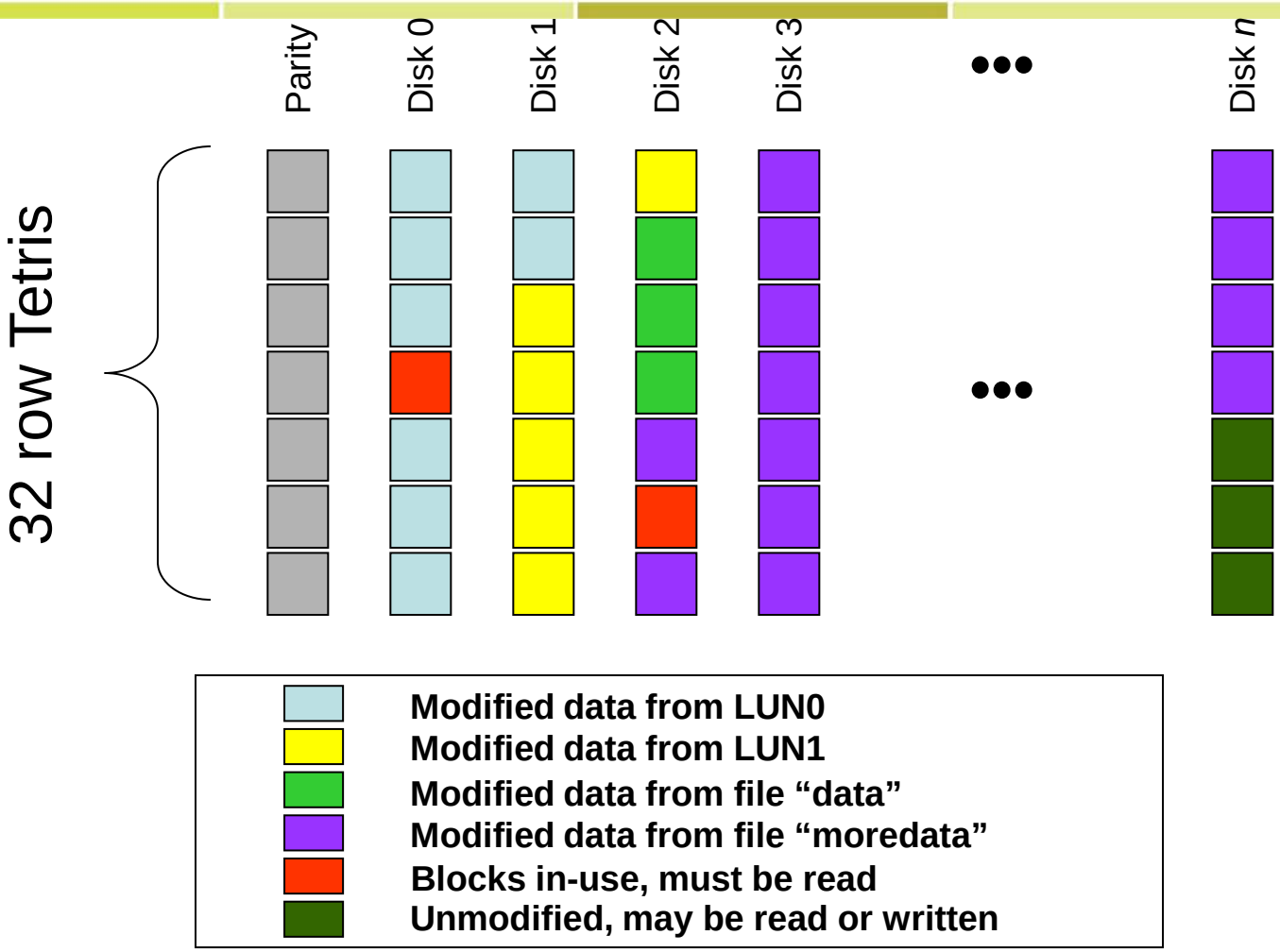
Shadow Paging File System Write



Shadow Paging FS After Write



How Random Writes are Optimized

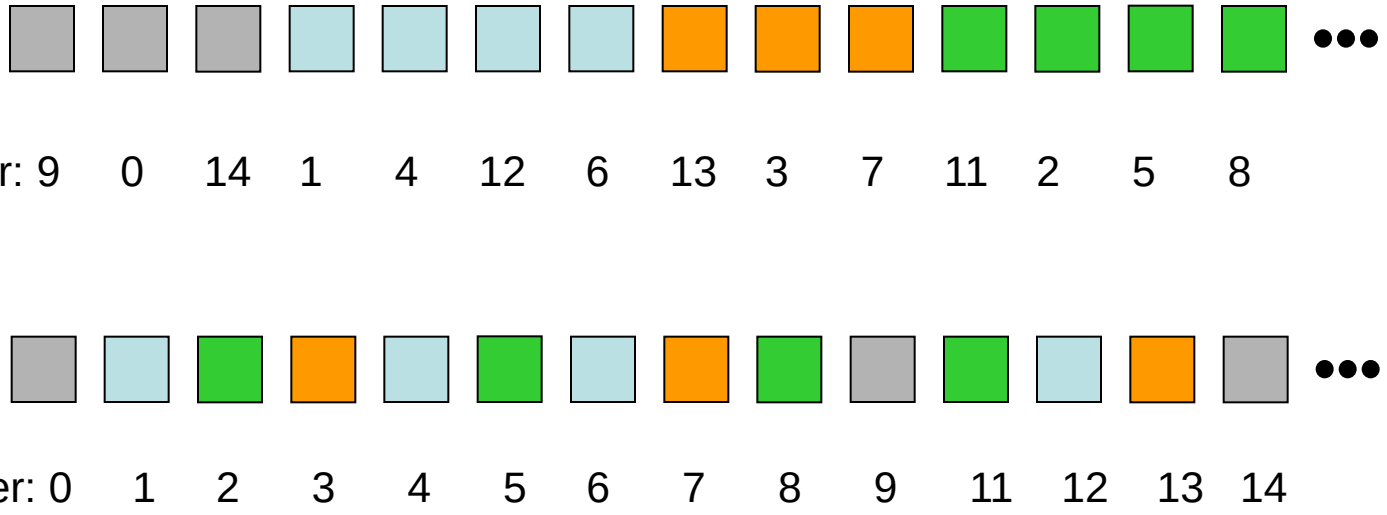


- Other write optimization methods are possible with shadow paging file systems.
- This example shows one write optimization method used by NetApp’s WAFL.

Temporal versus Spatial Locality

- Temporal Locality
  - Updates are written in batches
  - Each batch of updates are allocated at one time
  - All data written at one time will be close together
  - This algorithm works extremely well for purely random or purely sequential workloads

Sequential Read After Random Write



Problem Statement

- Problem
  - Suboptimal sequential read after random write performance
- Goals
  - Optimize sequential reads after random writes
  - Maintain random write optimizations
  - Maintain high-performance snapshots

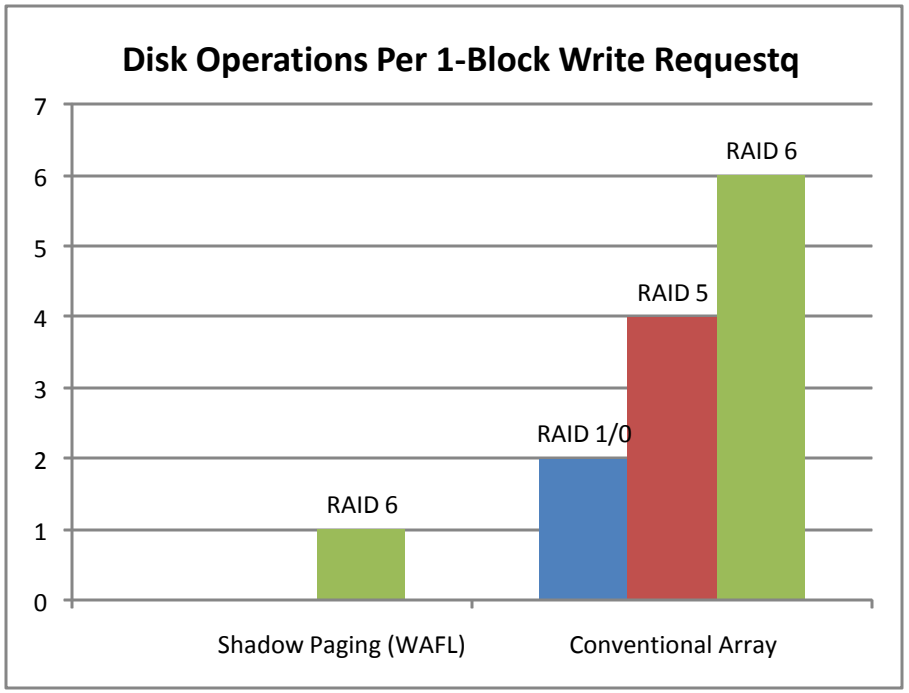
Optimization Technologies

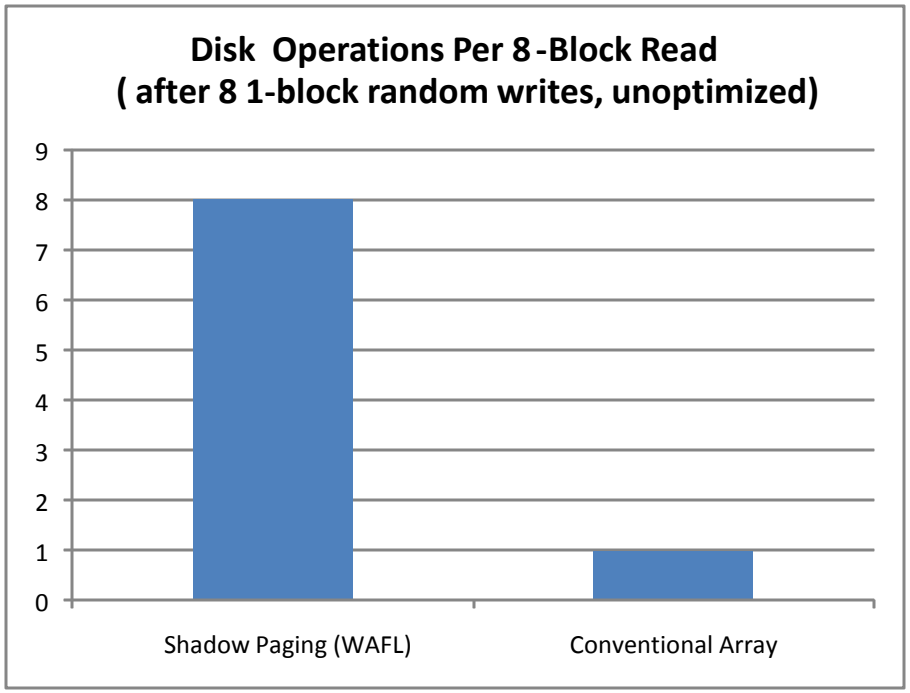
- Optimization without changing layout
  - Readahead
  - Dummy reads
- Write in place option for shadow paging file system
  - Doesn't satisfy goals
    - Poor random write performance
    - Poor snapshot performance (requires copy-out snapshots)

More Optimization Techniques

- Reallocate
  - Read file
  - Re-write blocks if layout can be improved
- Reallocate on read
  - Reallocate "on the fly"
    - Uses data read by the user
    - Optimizes only parts of file being used
  - Re-write blocks if layout can be improved

Analysis Background





Analysis of Reallocate on Read

File System Type	Random Write of n Blocks	First Read	Layout Optimization	Subsequent Reads	Write + One Read	Write + m Reads
Write in Place	2n	1	0	1	2n + 1	2n + m
Shadow Paging	< n	n	O(1)	1	<n + n + O(1) ≈ 2n + 1	2n + m

Assumptions:

- Write in Place file system uses RAID 1/0 (minimal write costs)
- Shadow Paging file system uses RAID 6 (better space utilization and protection)

Notes:

- The cost to write n blocks in a shadow paging file system varies by implementation.
- The "<n" in the above table was measured using WAFL and a variety of workloads.

Results: Write Optimizations

File System Type	SPC-1 IOPS per Disk Spindle
Write in Place (Conventional Array)	24997.49/154 =162
Shadow Paging (WAFL)	30985.90/140 = 221

- SPC-1 publication of two matched systems
  - SPC-1 is 60% writes; mostly random access
- Write in Place: 154 146GB 15K RPM disks with RAID 1/0
- Shadow Paging: 140 144GB 15K RPM disks with RAID 6
- Result: write optimization possible with shadow paging file systems

Reference to SPC-1 Full Disclosure Reports:
• http://www.storageperformance.org/results/a00057_NetApp_FAS3040_full-disclosure-r1.pdf
• http://www.storageperformance.org/results/a00059_NetApp_EMC-CX3-M40_full-disclosure-r1.pdf

Results: Reallocate after Read

Read	Average Physical Read Size Without Read Reallocation (KB)	Average Physical Read Size With Read Reallocation (KB)
1st	15	15
2nd	17	99

- 8 data spindles, 2 parity spindles (WAFL RAID-DP)
- Initialized with 8KB, 80% random, 60% write workload; tested with 64KB 100% read workload
- All workloads used 30 threads for 30 minutes
- Result: layout optimization possible with shadow paging file systems

Future Directions

- Increase the amount of data accessible in a single long read from disk
  - Requires improving quality of free space
  - Ongoing projects will clean free space
- Integration with SSDs
  - Shadow paging file systems are "flash friendly"
  - Optimize data on spinning disk for sequential access
  - Optimize data on flash for random access

Conclusion

- Shadow paging benefits are possible without sacrificing sequential read performance.
- When using a shadow paging file system on a parity protected RAID, a sequence of random write, optimize, and sequential read operations is cheaper than conventional write-in-place followed by sequential read.