DongJin Lee[1], Michael O'Sullivan[1], Cameron Walker[1]
Monique MacKensize[2]

[1]The University of Auckland
New Zealand

[2]The University of St Andrews
United Kingdom

Robust Benchmarking for Archival Storage Tiers

–PDSW 2011–

Motivation

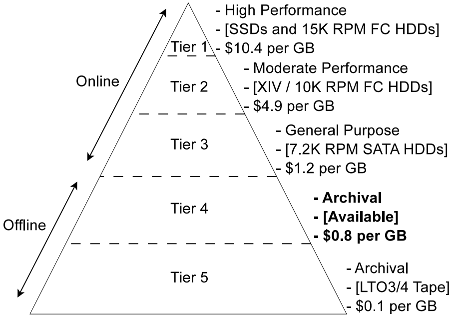
Storage Tiers

- Organizations use ‘tiered’ storage systems
- Low overall cost, high capacity and high performance
- Increasing amount of read/write request in recent years
- Studies on how to efficiently utilize and build better storage tier

Motivation

Storage Tiers

- Organizations use ‘tiered’ storage systems
- Low overall cost, high capacity and high performance
- Increasing amount of read/write request in recent years
- Studies on how to efficiently utilize and build better storage tier



Background and Introduction 1

Storage Design (Our research group)

- Build an optimized storage system (designing better node(s))
- Based on tier Requirements, e.g., cost($), capacity(TB), performance(MB/s) and power(W)
- Based on Architecture, e.g., file system
- Based on Component, e.g., disk-based, RAID, motherboard types, network types (commodity types)
- *Need to accurately measure MB/s using 'typical archival workload'*

Background and Introduction 1

Storage Design (Our research group)

- Build an optimized storage system (designing better node(s))
- Based on tier Requirements, e.g., cost($), capacity(TB), performance(MB/s) and power(W)
- Based on Architecture, e.g., file system
- Based on Component, e.g., disk-based, RAID, motherboard types, network types (commodity types)
- *Need to accurately measure MB/s using 'typical archival workload'*

Archival workload

- Important in designing/modeling for the archival storage system to meet the expected performance result, e.g.,
- How much MB/s gain do we observe when adding a certain number of disks?
- Would different workloads give different results?

Background and Introduction 2

Workload: access pattern

- What kind of workloads do archival tiers store/receive?
- What is the typical case? (need this to design the system)
- For archival tier: data migration and data retrieval

Background and Introduction 2

Workload: access pattern

- What kind of workloads do archival tiers store/receive?
- What is the typical case? (need this to design the system)
- For archival tier: data migration and data retrieval

Workload: file size

- Typical files experienced by the archival tier
- Characterize and model the file sizes
- Generate typical archival workload

Background and Introduction 2

Workload: access pattern

- What kind of workloads do archival tiers store/receive?
- What is the typical case? (need this to design the system)
- For archival tier: data migration and data retrieval

Workload: file size

- Typical files experienced by the archival tier
- Characterize and model the file sizes
- Generate typical archival workload

Observation

- Observe empirical file size distributions from the HPC sites[a]
- Develop models for file sizes with variations

[a] S. Dayal. *Characterizing HEC storage systems at rest*. Technical Report CMU-PDL-08-109, Carnegie Mellon University Parallel Data Lab, 2008.

Background and Introduction 3

Traditional workload

- Example tools: IOmeter, IOzone, Filebench, SPC-1
- Limited distribution-based workload and limited file testing
- No Archival-distribution workload

Background and Introduction 3

Traditional workload

- Example tools: IOmeter, IOzone, Filebench, SPC-1
- Limited distribution-based workload and limited file testing
- No Archival-distribution workload

Archival workload

- HSM write: batch file selection and migration (seq-write)
- HSM read: retrieval file access from multiple disks/nodes (rand-read)
- 'active' performance; no temporal access patterns (Discussion)
- Capacity utilization (total volume %) with distributions

Background and Introduction 3

Traditional workload

- Example tools: IOmeter, IOzone, Filebench, SPC-1
- Limited distribution-based workload and limited file testing
- No Archival-distribution workload

Archival workload

- HSM write: batch file selection and migration (seq-write)
- HSM read: retrieval file access from multiple disks/nodes (rand-read)
- 'active' performance; no temporal access patterns (Discussion)
- Capacity utilization (total volume %) with distributions

Archival workload

- Apply the archival file size distribution into a benchmark tool
- Measure the performance e.g., archival vs non-archival, archival vs traditional fixed files

Observed file sizes

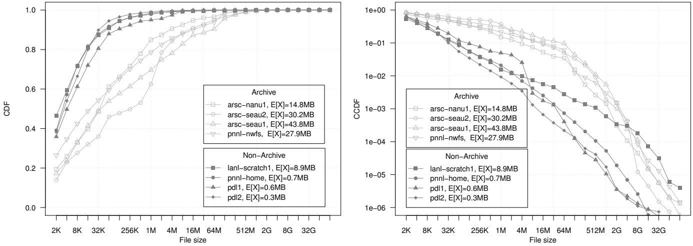
Empirical file size distribution from HPC

- Archive: `arsc-nanu1`, `arsc-seau2`, `arsc-seau1`, `pnnl-nwfs`
- 5.3M–13.7M files, 69TB–305TB volume
- Non-archive: `lanl-scratch1`, `pnnl-home`, `pdl1`, `pdl2`
- 1.5M–11.3M files, 1.2TB–9.2TB volume

Observed file sizes

Empirical file size distribution from HPC

- Archive: `arsc-nanu1`, `arsc-seau2`, `arsc-seau1`, `pnnl-nwfs`
  - 5.3M–13.7M files, 69TB–305TB volume
- Non-archive: `lanl-scratch1`, `pnnl-home`, `pdl1`, `pdl2`
  - 1.5M–11.3M files, 1.2TB–9.2TB volume



- Non-Archive: 61% <8KB and 81% <32KB (avg. 700KB)
- Archive: 28% <8KB and 36% <32KB (avg. 29.2MB)

Fitting file size distribution 1

Gamma and Gen. Gamma distribution

- $f(x;\theta,k,p) = \frac{(p/\theta^k)x^{k-1}e^{-(x/\theta)^p}}{\Gamma(k/p)}$, for $x \geq 0$, and $\theta, k, p > 0$
- Using `gnls` to find a parameter scale (θ) and shape (k,p)

Fitting file size distribution 1

Gamma and Gen. Gamma distribution

- $f(x;\theta,k,p) = \frac{(p/\theta^k)x^{k-1}e^{-(x/\theta)^p}}{\Gamma(k/p)}$, for $x \geq 0$, and $\theta, k, p > 0$
- Using `gnls` to find a parameter scale (θ) and shape (k,p)

Robustness of the fit

- We want to consider possible variability of the dataset
- Envelopes: risks/errors of typical file size distribution from the dataset
- Confidence Intervals: lower-bound and upper-bound
- i.e., more larger files and more smaller files

Fitting file size distribution 1

Gamma and Gen. Gamma distribution

- $f(x; \theta, k, p) = \frac{(p/\theta^k) x^{k-1} e^{-(x/\theta)^p}}{\Gamma(k/p)}$, for $x \geq 0$, and $\theta, k, p > 0$
- Using `gnls` to find a parameter scale (θ) and shape (k,p)

Robustness of the fit

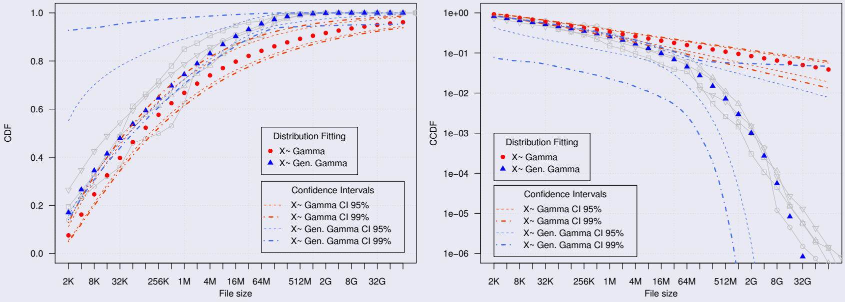
- We want to consider possible variability of the dataset
- Envelopes: risks/errors of typical file size distribution from the dataset
- Confidence Intervals: lower-bound and upper-bound
- i.e., more larger files and more smaller files

CI Bootstrapping

- bootstrapped CDFs $F_i^B(x)$, each parameter $(\theta_i^B, k_i^B, p_i^B), i = 1, \ldots, N$
- Sort the $F_i^B(x)$ to find percentiles, i.e., 95th and 99th
- Identify lower-bound $\frac{\alpha}{2}$ and upper-bound $1 - \frac{\alpha}{2}$

Fitting file size distribution 2

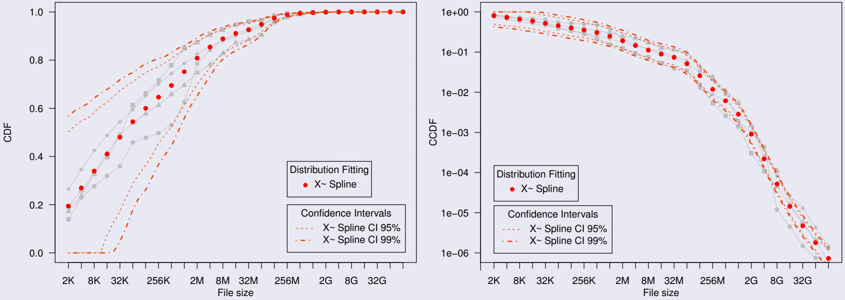
Gamma and Gen. Gamma distribution



- Gamma: CDF good-fit at the body, poor-fit at the tail
- Gen. Gamma: good-fit at the body, good-fit at the tail
- Both distribution functions produced poor CIs.
- e.g., produced large probabilities of files with >64MB
- lower-bound (E[X]=1.7GB) and upper-bound (E[X]=3.8MB)

Fitting file size distribution 3

Spline distribution



- Set of piecewise polynomials joining ‘knot’ points of the overall function
- We made sure to use a monotonically non-decreasing function
- Using `gnls` to find a best coefficient for each piece

Generating a typical workload

Fileset

- Convert CDF to PDF and using either 1) file counts or 2) volume
- A CDF $F(x) = \Pr(X \leq x)$ to $F(x) = \Pr(X = x)$
- $\Pr(X = 4\text{KB}) = \Pr(X = x_2) = F(x_2) - \Pr(X = 2\text{KB})$, and so on for $\Pr(X = x_i), i \geq 2$.
- Produce 3 filesets (file size PDFs: lower-, median- and upper-bound)
- e.g., a fileset with C files (e.g., 50k), or fileset with V (e.g., 2.4TB)

Generating a typical workload

Fileset

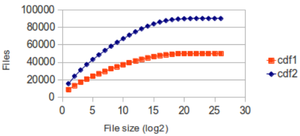
- Convert CDF to PDF and using either 1) file counts or 2) volume
- A CDF $F(x) = \Pr(X \leq x)$ to $F(x) = \Pr(X = x)$
- $\Pr(X = 4\text{KB}) = \Pr(X = x_2) = F(x_2) - \Pr(X = 2\text{KB})$, and so on for $\Pr(X = x_i), i \geq 2$.
- Produce 3 filesets (file size PDFs: lower-, median- and upper-bound)
- e.g., a fileset with C files (e.g., 50k), or fileset with V (e.g., 2.4TB)

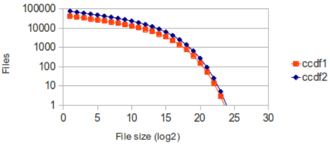
Example (FFSB tool)

```
size_weight   2KB   15322
size_weight   4KB    8609
size_weight   8KB    7132
...
size_weight   1GB     382
size_weight   2GB     176
size_weight   4GB     665
```

Example of a fileset size

			Files (C)			Volume (V) GB	
			50000			2400	
File Size (KB)		PDF	file (c)	volume (v)	v coeff	volume (v)	file (c)
2KB	2	*0.1699179*	**8496**	16991.79	0.3398358	30645	**15322**
4KB	4	*0.0954718*	**4774**	19094.36	0.3818872	34437	**8609**
8KB	8	*0.0790857*	**3954**	31634.28	0.6326856	57053	**7132**
16KB	16	*0.0700224*	**3501**	56017.92	1.1203584	101029	**6314**
32KB	32	*0.0639579*	**3198**	102332.64	2.0466528	184558	**5767**
64KB	64	*0.0594704*	**2974**	190305.28	3.8061056	343218	**5363**
128KB	128	*0.0559066*	**2795**	357802.24	7.1560448	645300	**5041**
256KB	256	*0.0528908*	**2645**	677002.24	13.540045	1220981	**4769**
512KB	512	*0.0501611*	**2508**	1284124.16	25.682483	2315932	**4523**
1MB	1024	*0.0475013*	**2375**	2432066.56	48.641331	4386259	**4283**
2MB	2048	*0.0447106*	**2236**	4578365.44	91.567309	8257133	**4032**
4MB	4096	*0.0415972*	**2080**	8519106.56	170.38213	15364303	**3751**
8MB	8192	*0.0379878*	**1899**	15559802.88	311.19606	28062276	**3426**
16MB	16384	*0.0337579*	**1688**	27654471.68	553.08943	49875145	**3044**
32MB	32768	*0.028879*	**1444**	47315353.60	946.30707	85333763	**2604**
64MB	65536	*0.0234704*	**1174**	76907806.72	1538.1561	138704079	**2116**
128MB	131072	*0.0178329*	**892**	116869693.44	2337.3939	210775784	**1608**
256MB	262144	*0.0124262*	**621**	162872688.64	3257.4538	293742694	**1121**
512MB	524288	*0.0077618*	**388**	203470929.92	4069.4186	366962071	**700**
1GB	1048576	*0.0042314*	**212**	221847224.32	4436.9445	400103921	**382**
2GB	2097152	*0.0019516*	**98**	204640092.16	4092.8018	369070668	**176**
4GB	4194304	*0.0007346*	**37**	154056785.92	3081.1357	277843116	**66**
8GB	8388608	*0.0002167*	**11**	90890567.68	1817.8114	163922143	**20**
16GB	16777216	*0.0000478*	**2**	40097546.24	801.95092	72316368	**4**
32GB	33554432	*0.0000075*	**0**	12582912.00	251.65824	22693421	**1**
64GB	67108864	*0.0000007*	**0**	2348810.24	46.976205	4236105	**0**
			Files (C)	**Volume (V) GB**		**Volume (V) GB**	**Files (C)**
		1	50000	1331		2400	90176





Performance Comparisons 1

Benchmarking

- Archival vs Non-archival (empirical/model distributions)
- Archival vs fixed file size (e.g., 128KB, 1MB, 4MB)
- Consistent filesets with increasing storage capacity utilization

Performance Comparisons 1

Benchmarking

- Archival vs Non-archival (empirical/model distributions)
- Archival vs fixed file size (e.g., 128KB, 1MB, 4MB)
- Consistent filesets with increasing storage capacity utilization

Test setup

- Intel CPU Xeon 5630 (2.53Ghz), 18GB RAM, Intel X58/5520 Chipset
- 12TB – 6×2TB WDC `WD20EAR`, LSI 2108 RAID Controller (512MB)
- LSI 2108 RAID Controller (512MB) RAID 0 write-through mode 8K directIO
- Filesystem: Local ext4, and Ceph using btrfs and ext4
- Ceph: 2 machines: one client (workload generator), one CMDS/CMON/COSD
- Bonded 4×Gb/s Intel Eth NIC (*iperf* measurement - 3.4Gb/s)

Performance Comparisons 2

Step procedure

1. Filesets: 1%, 5%, 20% and 40% capacity utilizations
2. Sequential-write the entire fileset
3. Random-read from that fileset (128, 256 and 512 threads) min. 30m
4. Repeat: recreate the partition, drop all caches between the steps

Performance Comparisons 2

Step procedure

1. Filesets: 1%, 5%, 20% and 40% capacity utilizations
2. Sequential-write the entire fileset
3. Random-read from that fileset (128, 256 and 512 threads) min. 30m
4. Repeat: recreate the partition, drop all caches between the steps

Overall observations amongst setups

- sequential-write: 450–500MB/s local ext4, 70–80MB/s Ceph
- No obvious performance differences for the writes, and random-read threads

Performance Comparisons 2

Step procedure

1. Filesets: 1%, 5%, 20% and 40% capacity utilizations
2. Sequential-write the entire fileset
3. Random-read from that fileset (128, 256 and 512 threads) min. 30m
4. Repeat: recreate the partition, drop all caches between the steps

Overall observations amongst setups

- sequential-write: 450–500MB/s local ext4, 70–80MB/s Ceph
- No obvious performance differences for the writes, and random-read threads

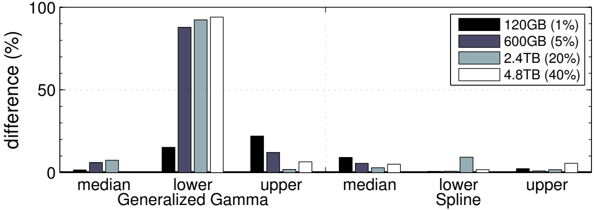
Random-read

- Archival vs Non-archival: large performance difference
- For example, at 5% fileset (600GB)
- Archivals: 39.5MB/s vs. Non-archivals: 27.3MB/s (31% difference)

Result 1 (ext4)

Capacity Utilization	Empirical archival distributions					Fitted models: Generalized Gamma			Fitted models: Spline		
	arsc-nanu1 $E[X] = 14.8$MB	arsc-seau2 $= 30.2$MB	arsc-seau1 $= 43.8$MB	pnnl-nwfs $= 27.9$MB	avg. $= 29.2$MB	median $= 24.5$MB	lower $= 1.7$GB	upper $= 3.8$MB	median $= 25.8$MB	lower $= 28.7$MB	upper $= 8.1$MB
120GB (1%)	55.4	58.3	69.8	58.7	60.6	61.5	51.3	47.2	66.1	60.1	59.1
600GB (5%)	42.3	35.9	43.6	36.2	39.5	41.9	4.8	34.7	41.7	39.8	39.9
2.4TB (20%)	35.9	32.9	41.3	31.2	35.3	32.7	2.7	36.0	34.3	38.6	34.7
4.8TB (40%)	31.1	37.6	36.8	29.7	33.8	33.8	2.0	36.0	35.5	33.2	31.9

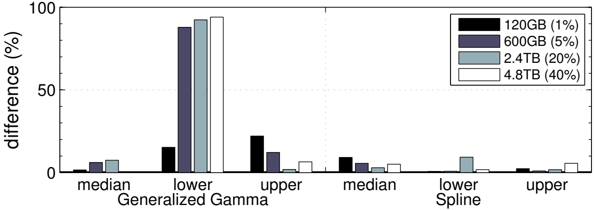
Table: Random-read MB/s of empirical archival distributions and fitted models



Result 1 (ext4)

Capacity Utilization	Empirical archival distributions: arsc-nanu1 E[X]=14.8MB	arsc-seau2 =30.2MB	arsc-seau1 =43.8MB	pnnl-nwfs =27.9MB	avg. =29.2MB	Fitted models: Generalized Gamma median =24.5MB	lower =1.7GB	upper =3.8MB	Spline median =25.8MB	lower =28.7MB	upper =8.1MB
120GB (1%)	55.4	58.3	69.8	58.7	60.6	61.5	51.3	47.2	66.1	60.1	59.1
600GB (5%)	42.3	35.9	43.6	36.2	39.5	41.9	4.8	34.7	41.7	39.8	39.9
2.4TB (20%)	35.9	32.9	41.3	31.2	35.3	32.7	2.7	36.0	34.3	38.6	34.7
4.8TB (40%)	31.1	37.6	36.8	29.7	33.8	33.8	2.0	36.0	35.5	33.2	31.9

Table: Random-read MB/s of empirical archival distributions and fitted models

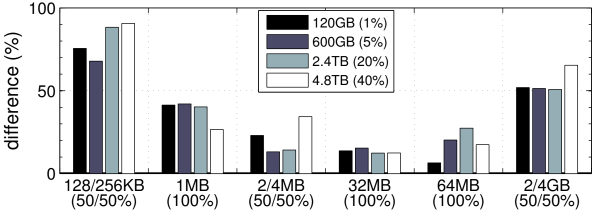


- Increasing capacity utilization decreases the performance
- Fileset for median generally followed close to the empirical archivals
- Gen. Gamma's lower-bound performance deteriorates

Result 2 (ext4)

Cap. Util.	Fixed file size model 128/256KB (50/50%)	1MB (100%)	2/4MB (50/50%)	32MB (100%)	64MB (100%)	2/4GB (50/50%)
1%	14.8	35.5	46.6	52.2	56.6	92.0
5%	12.7	22.9	34.3	45.6	47.5	19.2
20%	4.1	21.1	30.3	39.7	45.0	17.4
40%	3.2	24.8	22.2	38.0	39.7	11.7

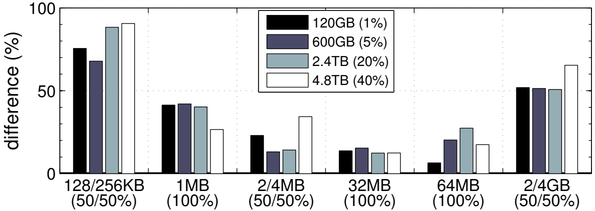
Table: Random-read MB/s of fixed file size models



Result 2 (ext4)

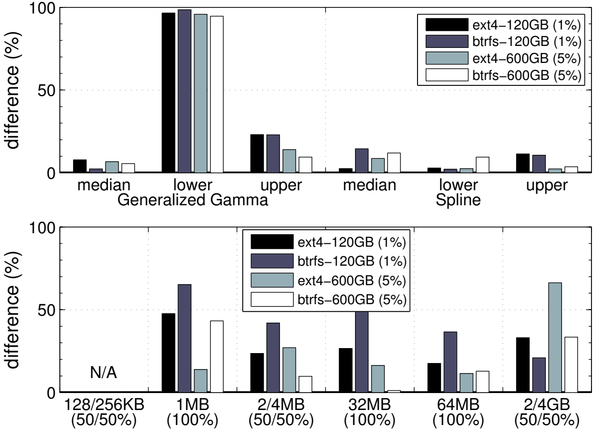
Cap. Util.	Fixed file size model 128/256KB (50/50%)	1MB (100%)	2/4MB (50/50%)	32MB (100%)	64MB (100%)	2/4GB (50/50%)
1%	14.8	35.5	46.6	52.2	56.6	92.0
5%	12.7	22.9	34.3	45.6	47.5	19.2
20%	4.1	21.1	30.3	39.7	45.0	17.4
40%	3.2	24.8	22.2	38.0	39.7	11.7

Table: Random-read MB/s of fixed file size models



- Fixed file size shows poor representation (large % difference)
- Closest are the 32MB fixed file size
- Coincident (large file sizes, e.g., 64MB, 2/4GB have different MB/s)

Result 3 (Ceph)



- Similar results to the local-ext4
- No obvious trend amongst the fixed file sizes
- i.e., 2/4MB, 32MB, 64MB files

Summary

Result summary

- Archival distributions are unique and produce different performance results; we use this workload to design the archival storage system
- Different disks/filesystems have different behaviors for a particular size
- Workloads are ran for a long period and with a large volume
- Upper- lower-bounds' performance did not differ much
- - small files do not 'show well'; need to test for much smaller filesets
- - possible to cut-off at a certain file size, e.g., 64MB and ignore the rest

Summary

Result summary

- Archival distributions are unique and produce different performance results; we use this workload to design the archival storage system
- Different disks/filesystems have different behaviors for a particular size
- Workloads are ran for a long period and with a large volume
- Upper- lower-bounds' performance did not differ much
- - small files do not 'show well'; need to test for much smaller filesets
- - possible to cut-off at a certain file size, e.g., 64MB and ignore the rest

Conclusion

- Distribution-based file size benchmarking for archival storage
- Robust envelopes considered for the observed empirical archives
- Workload generated, benchmarked and measured performance
- Accurate performance representation

Discussion

Assumptions

- Usage 'time of the day' (peak vs off-peak period)
- Dynamic reads and writes, actual access pattern
- Locality of the files and de-duplication

Thank you for attendances

Thanks

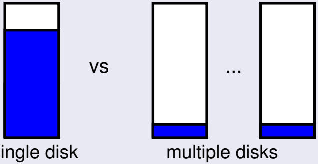
Anonymous feedbacks from the reviewers

Q&A

dongjin.lee@auckland.ac.nz
michael.osullivan@auckland.ac.nz
cameron.walker@auckland.ac.nz
monique@mcs.st-and.ac.uk
http://twiki.esc.auckland.ac.nz/twiki/bin/view/NDSG/WebHome

Additional (Fileset % capacity utilization)

no % fileset volume (capacity utilization)



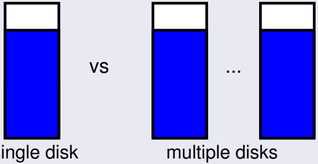
Example:

10% of 2TB disk (200GB fileset)

10x2TB disk (200GB fileset)

Each disk receives 20GB workload (less workload)

% fileset volume (capacity utilization)



Example:

10% of 2TB disk (200GB fileset)

10% of 10x2TB disk (2TB fileset)

Each disk receives 200GB workload (similar workload)