Bringsel: A Tool for Measuring Storage System Reliability, Uniformity, Performance and Scalability

John Kaitschuck

Cray Federal

CUG2007

jkaitsch@cray.com

5/2007

Overview

- Challenges in File Systems Testing and Technology

- Points for Consideration

- A Generalized Requirement Framework

- Bringsel, Yet Another File System Benchmark?

- Features

- Examples

- Sample Output

- Testing/Taxonomy

- Some Results

- Possible Future Directions for Bringsel

- Questions

Challenges in File System Testing and Technology

> "If seven maids with seven mops
> Swept it for half a year,
> Do you suppose," the Walrus said,
> "That they could get it clear?"
> -- Lewis Carroll

- Primary focus within community, users and suppliers.
- Rarely consider reliability (implied/assumed).
- Pace of hardware technology vs. system software.
- Limits on testing, temporal and hardware wise.
- Focus derived from RFP/SOW/Facility breakdown.
- Scaling, doing end to end testing.
- Historical context, past vs. present.
- Differing customer/user requirements.
- Sometimes ideas ignore operational context.

Points for Consideration

Partial

[1] Service Specifics - API's, Documentation, Security...

[2] Reliability - Given N bits, reflect N bits of content...

[3] Uniformity - Under load X, for period T...

[4] Performance - Provide high bandwidth, low latency...

[5] Scalability - Provide 1 -> 4 at sizes required...

Full

A Generalized Requirement Framework

$$\sum Se_a + \sum \mathrm{Re}_b + \sum Un_c + \sum Pe_d + \sum Sc_e$$

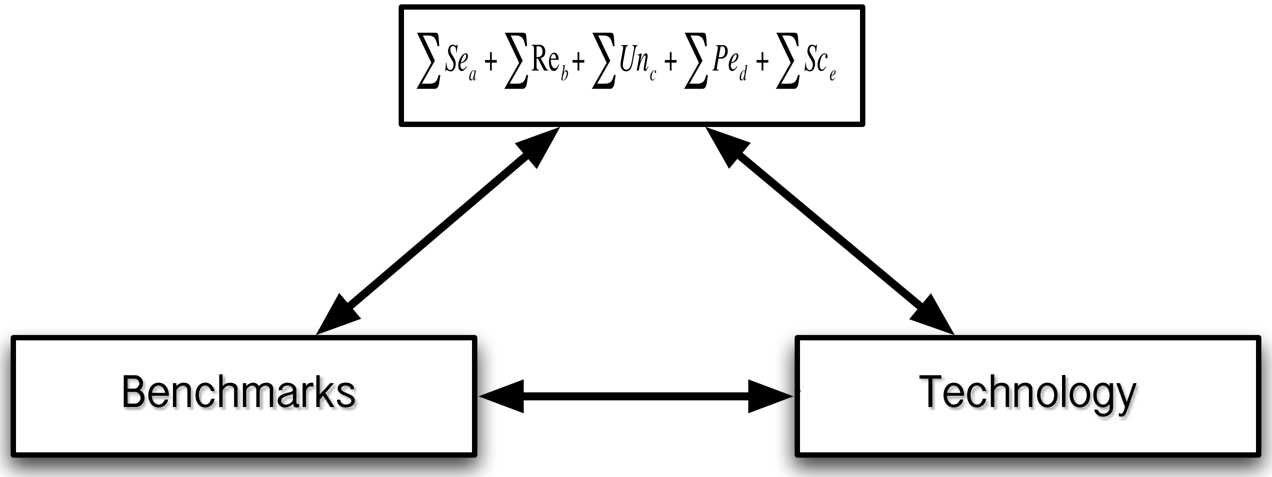
- Where these elements take on a series of unique values, which are...

  - Defined by the facility.

  - Defined by the application(s).

  - Constrained by the technology/architecture (fs, dfs, pfs).

A Generalized Requirement Framework: Ideally


$\sum Se_a + \sum \mathrm{Re}_b + \sum Un_c + \sum Pe_d + \sum Sc_e$
Benchmarks
Technology

Bringsel, Yet Another File System Benchmark?

- Plenty of existing benchmarks/utilities...
  bonnie++, iozone, filebench, perf, pdvt, ior, xdd, explode
  trace, etc.

- Not all are "operational inclusive" (mixed ops and blocks).

- Most focus on separated MD/Data testing.

- Need a known context, bringsel development started in
  ~1998, focused on HPTC, a strictly part time project.

- Need to have a code that is easy to modify,
  comment, extend, maintain and balance simplicity/complexity.

- Need a code with a known utilization history.
  (Industry, NSF, other Federal sites)

- Need to focus on central point within user space for "nd" I/O.

- Unique tools, enable unique discoveries.

- Diversification of available test programs.

Features

- Symmetric tree creation and population.
- MultiAPI support:
  POSIX, STREAM, MMAP, MPI_IO
- POSIX threads support (AD).
- File checksums via haval.
- Directory walks, across created structures.
- Metadata loop measurements.
- MSI support via MPI (MPP/Clusters).
- Mixed access types (RW, SR, etc.).
- Mixed block sizes (16K, 1024K, etc.).
- Remedial configuration file parsing.
- Coordinated looping/iteration support.
- Misc functionality:
  truncation, async I/O, appending, etc.
- Numerous reliability checks.
- Of course, Bandwidth and IOPS performance measurement as well.

Examples

Simple CLI Invocation

General File Operation

```
bringsel -T 4  -D /snarf/foo:1,2,2 -M -L -c -b 32 -S 100M alpha
```

Directory Walk

```
bringsel -T 4 -a sx -D /snarf/foo:1,2,2 -L
```

Examples

Configuration File Utilization

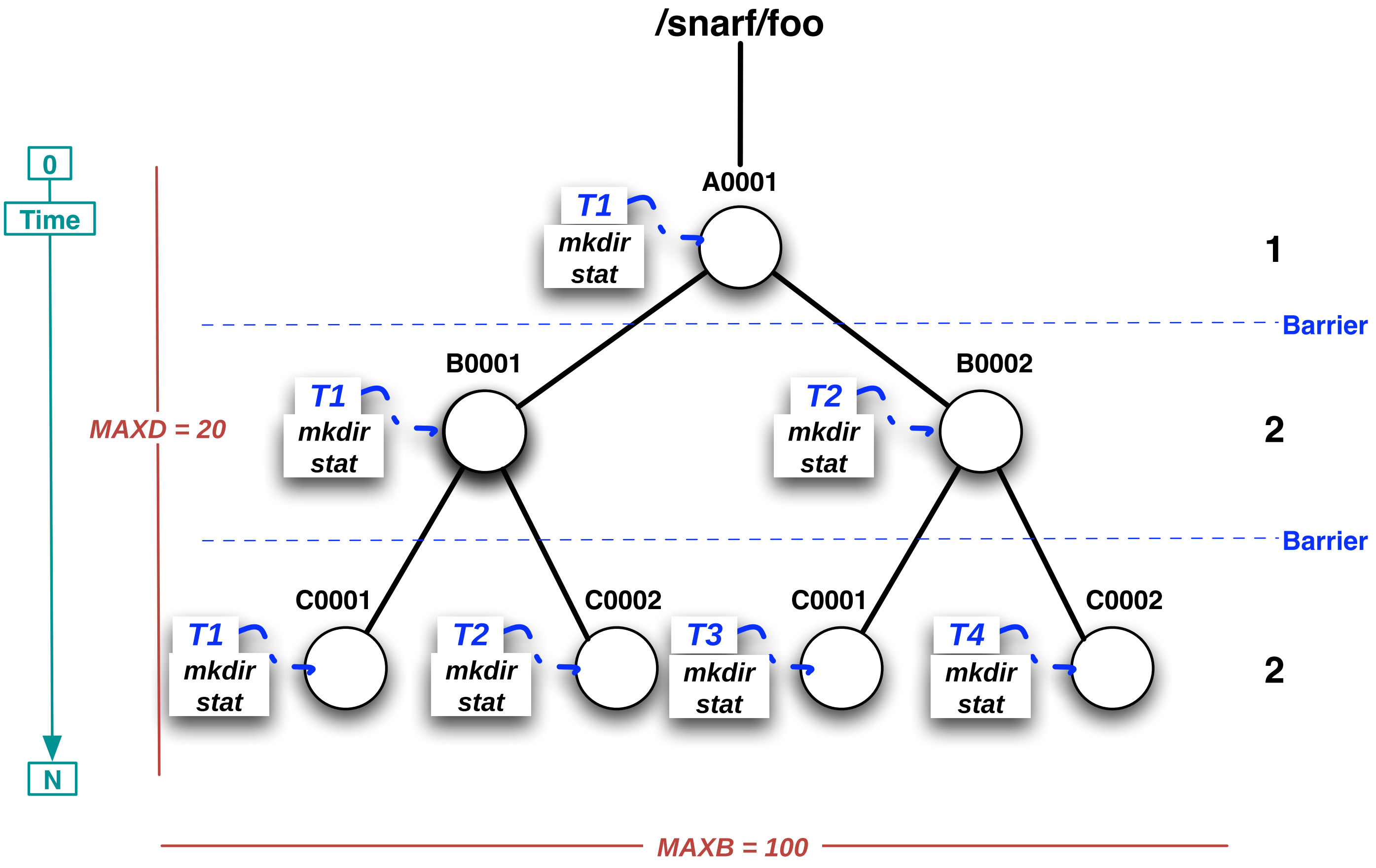
```
#
#      Comments begin with "#"
#
-T 4  -D /snarf/foo:1,2,2 -M -L -c -b 32 -S 100M alpha
-T 4 -a sx -D /snarf/foo:1,2,2 -L
```

Invocation

```
bringsel -C ./sample.cnf
```

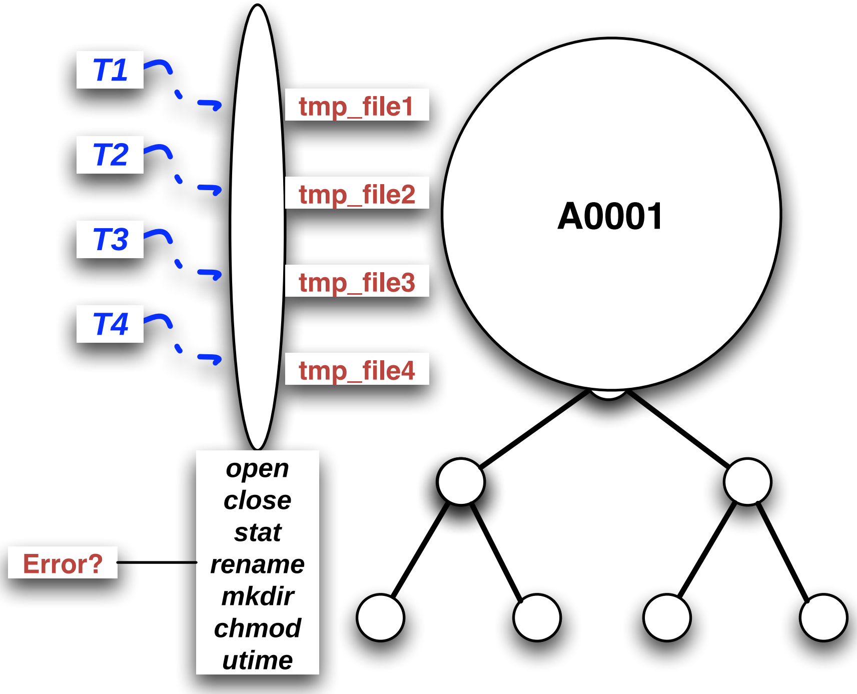
Example: Parallel Directory Creation

```
bringsel -T 4 -D /snarf/foo:1,2,2 -M -L -c -b 32 -S 100M alpha
```



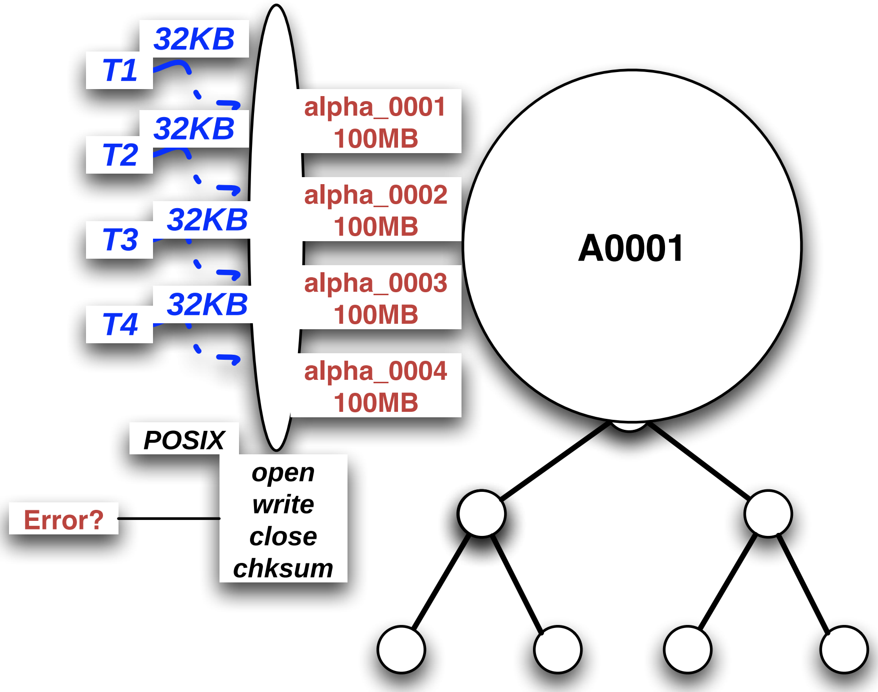
Example: Metadata Loop Operations

```
bringsel -T 4 -D /snarf/foo:1,2,2 -M -L -c -b 32 -S 100M alpha
```



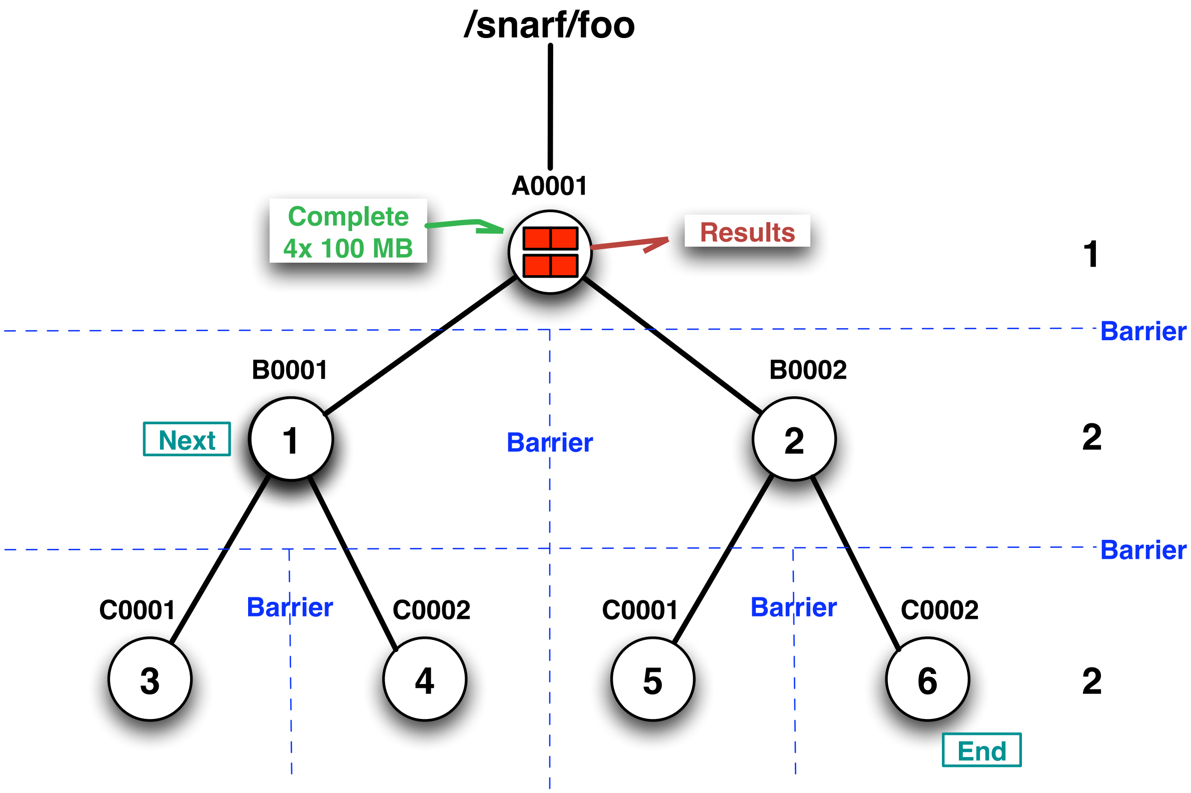
Example: File Operations

bringsel **-T 4** -D /snarf/foo:1,2,2 -M -L **-c -b 32 -S 100M alpha**



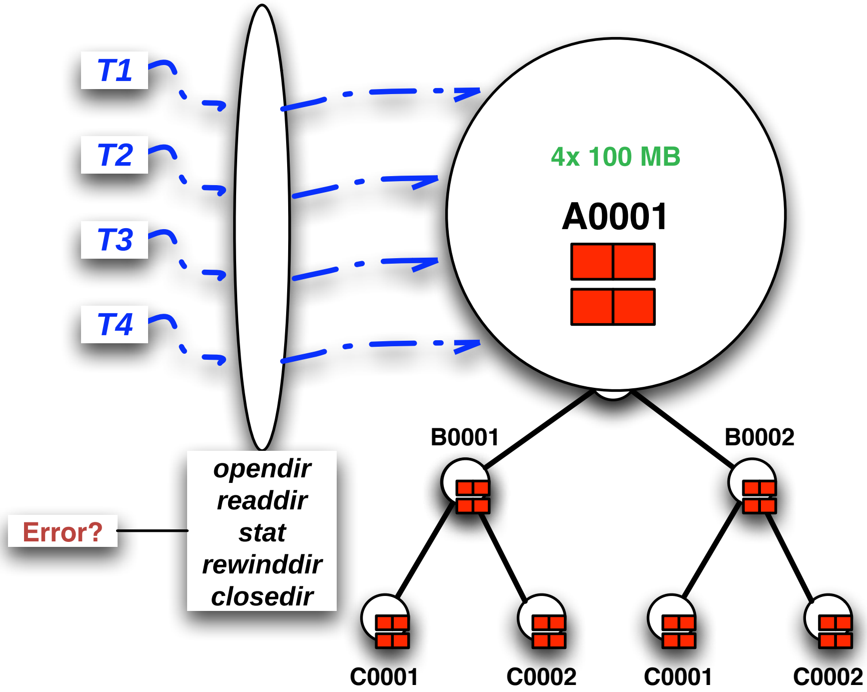
Example: Sequence of Operations

bringsel -T 4 -D /snarf/foo:1,2,2 -M -L -c -b 32 -S 100M alpha

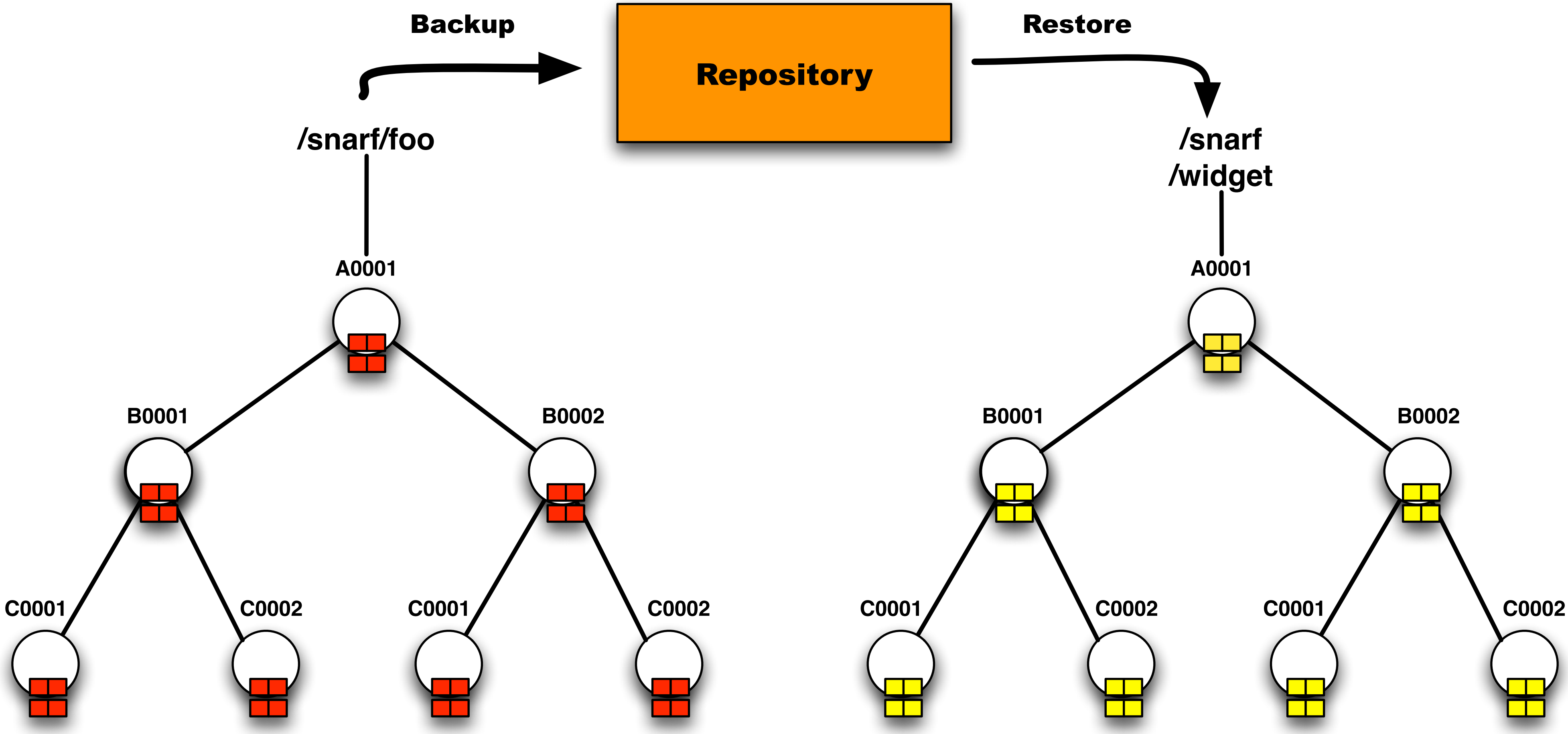


Example: Directory Walk

bringsel **-T 4 -a sx -D /snarf/foo:1,2,2** -L

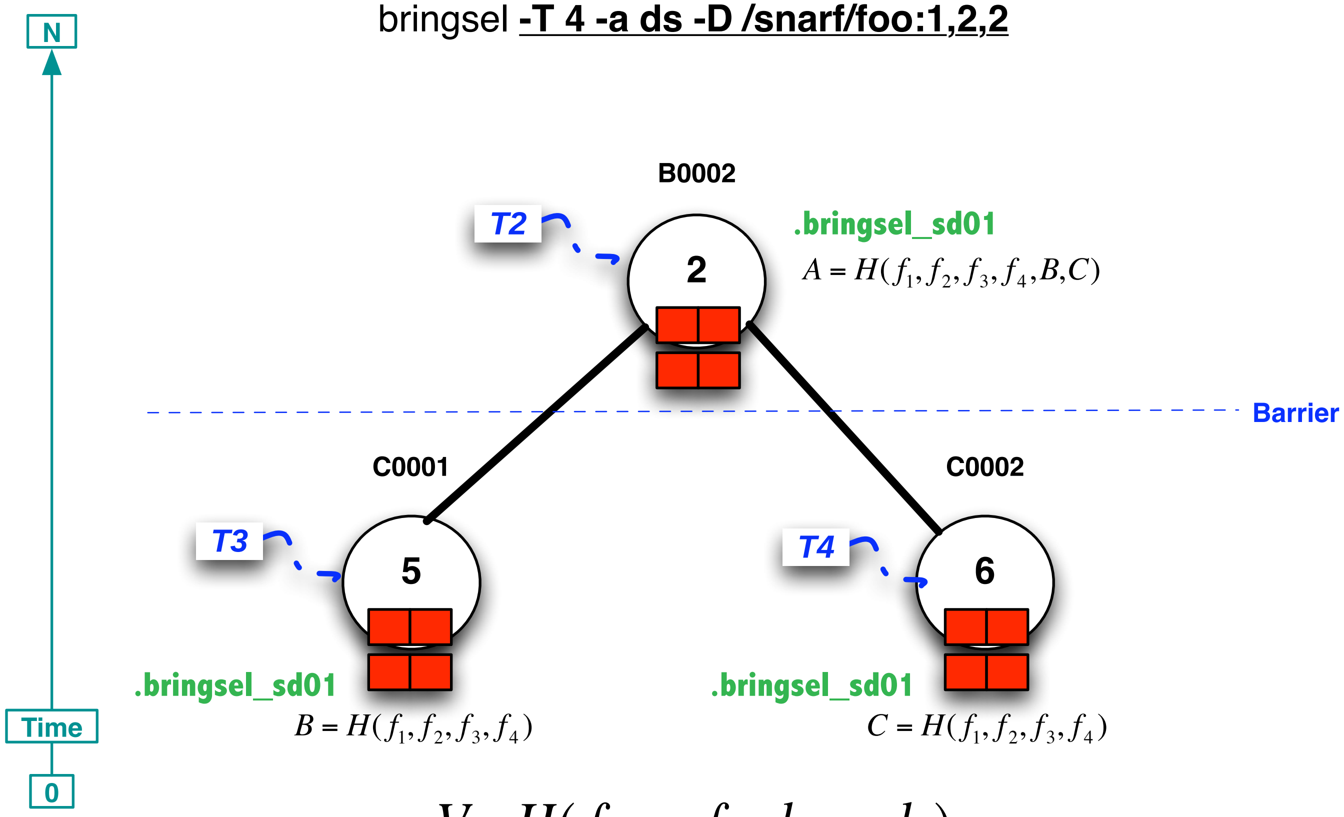


Example: Hash Trees



Example: Hash Tree Formulation

bringsel **-T 4 -a ds -D /snarf/foo:1,2,2**

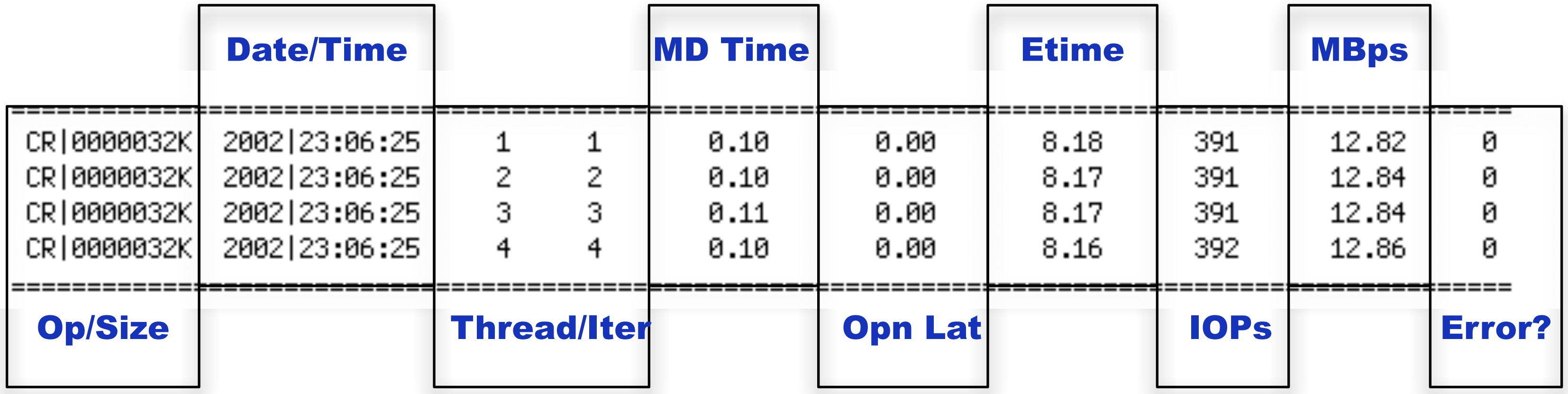

N
B0002
T2
2
.bringsel_sd01
$A = H(f_1, f_2, f_3, f_4, B, C)$
Barrier
C0001
C0002
T3
5
T4
6
.bringsel_sd01
$B = H(f_1, f_2, f_3, f_4)$
.bringsel_sd01
$C = H(f_1, f_2, f_3, f_4)$
Time
0

$$V = H(f_1 \rightarrow f_n, d_1 \rightarrow d_n)$$

$$H() \rightarrow SHA-256$$

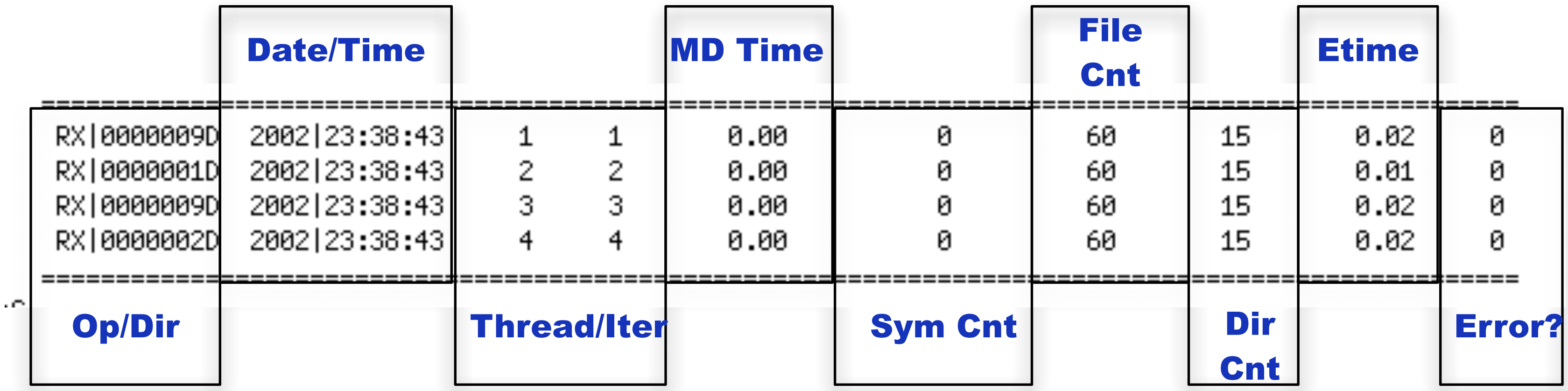
Sample Raw Output

Standard File Operations



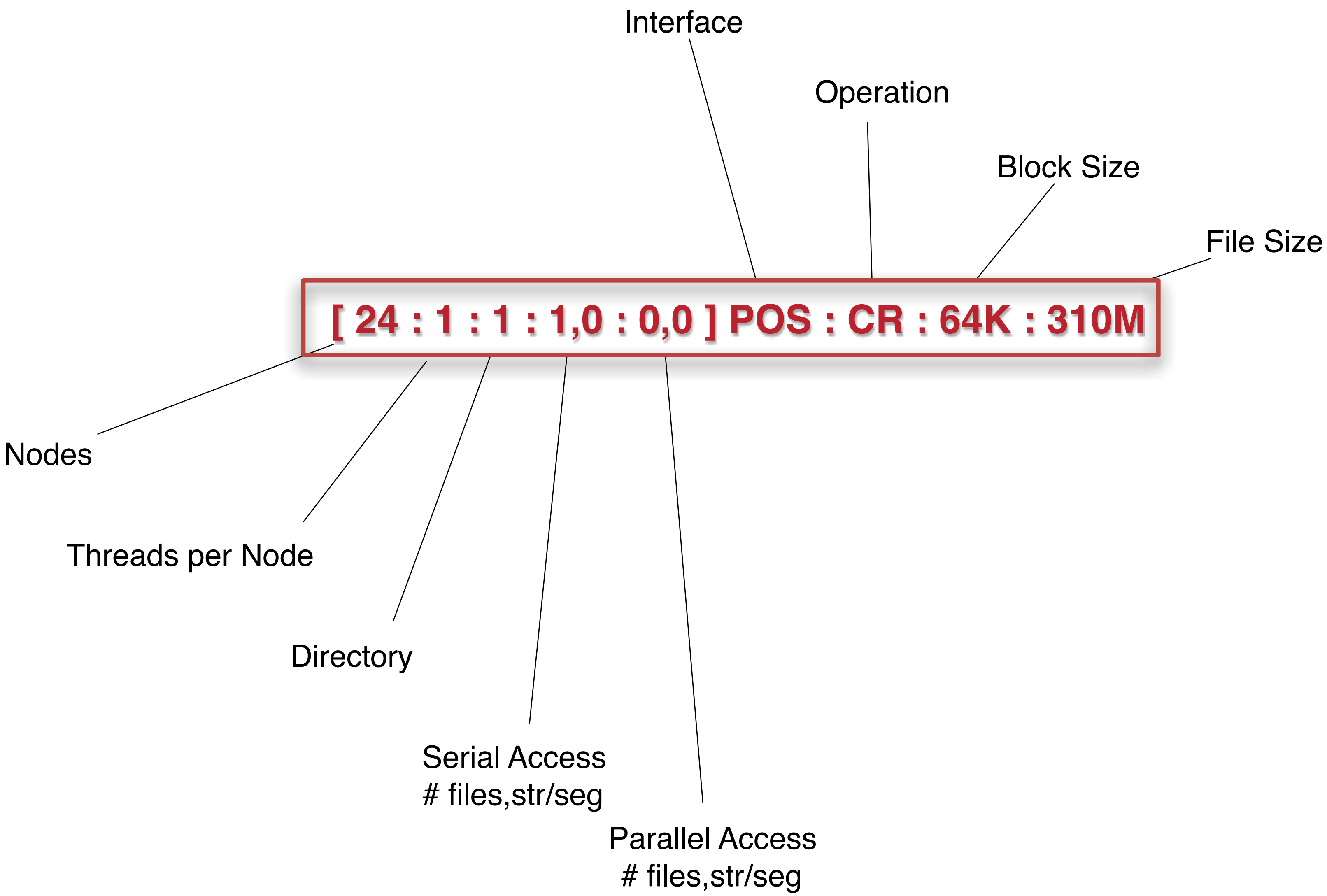
```
=================================================================================================
 CR|0000032K  2002|23:06:25    1     1     0.10     0.00     8.18    391    12.82    0
 CR|0000032K  2002|23:06:25    2     2     0.10     0.00     8.17    391    12.84    0
 CR|0000032K  2002|23:06:25    3     3     0.11     0.00     8.17    391    12.84    0
 CR|0000032K  2002|23:06:25    4     4     0.10     0.00     8.16    392    12.86    0
=================================================================================================
```

Directory Walk



```
=================================================================================================
 RX|0000009D  2002|23:38:43    1     1     0.00       0     60     15     0.02    0
 RX|0000001D  2002|23:38:43    2     2     0.00       0     60     15     0.01    0
 RX|0000009D  2002|23:38:43    3     3     0.00       0     60     15     0.02    0
 RX|0000002D  2002|23:38:43    4     4     0.00       0     60     15     0.02    0
=================================================================================================
```

Testing/Taxonomy



[24 : 1 : 1 : 1,0 : 0,0] POS : CR : 64K : 310M

Interface

Operation

Block Size

File Size

Nodes

Threads per Node

Directory

Serial Access
# files,str/seg

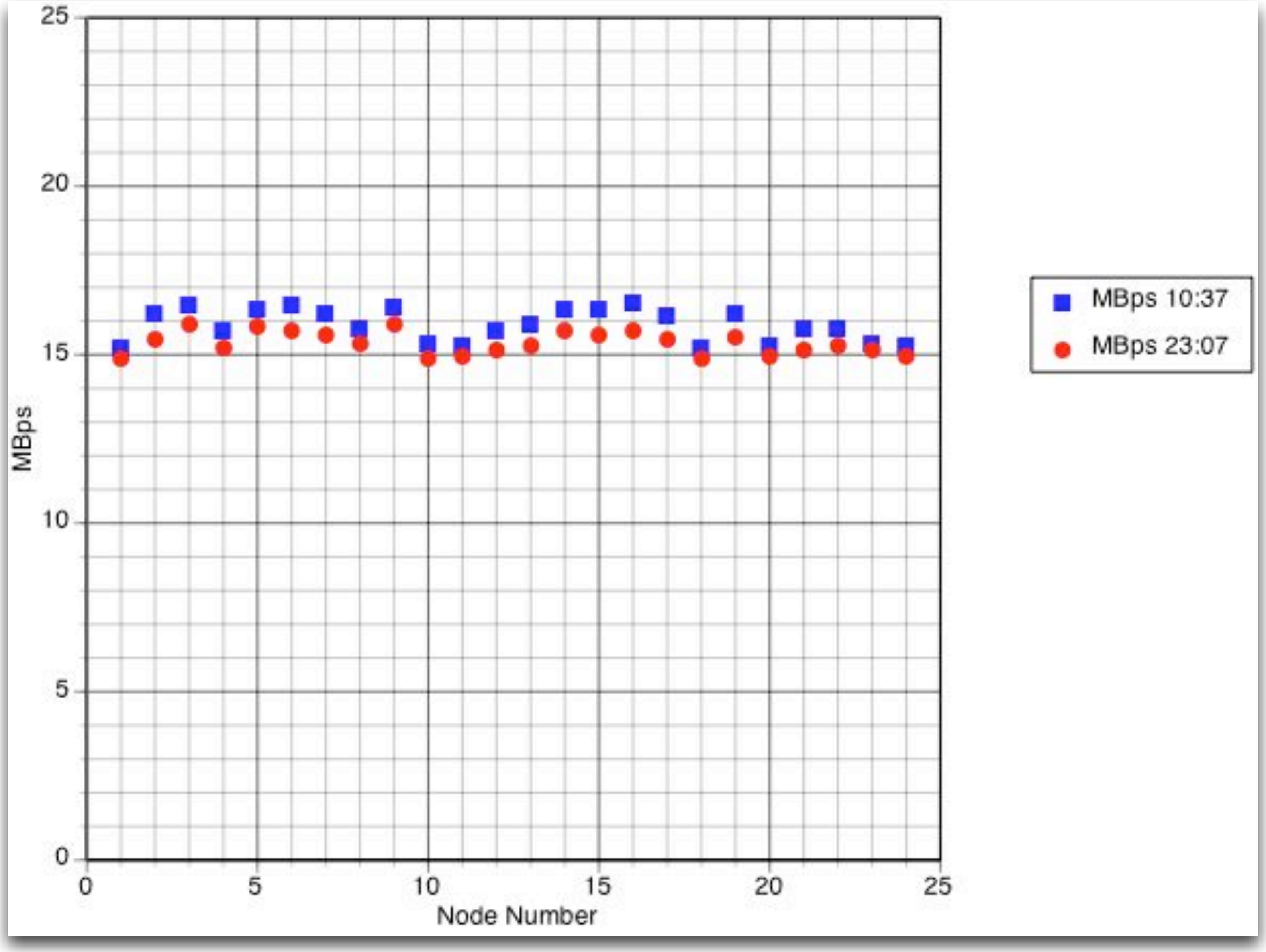
Parallel Access
# files,str/seg

Sample Results: Reliability

- Of 25 Tests...
  - ~350 TB of data written without corruption or access failures.
  - No major hardware failures in ~90 days of operation.
  - All checksums valid.
  - Early SLES9 NFS client problems under load, detected and corrected via patch. (735130)
  - 1 FC DDU failure, without data loss.
  - Spatial use from 0% to 100%+ during various test cases.
  - Test case durations of several minutes to several days.

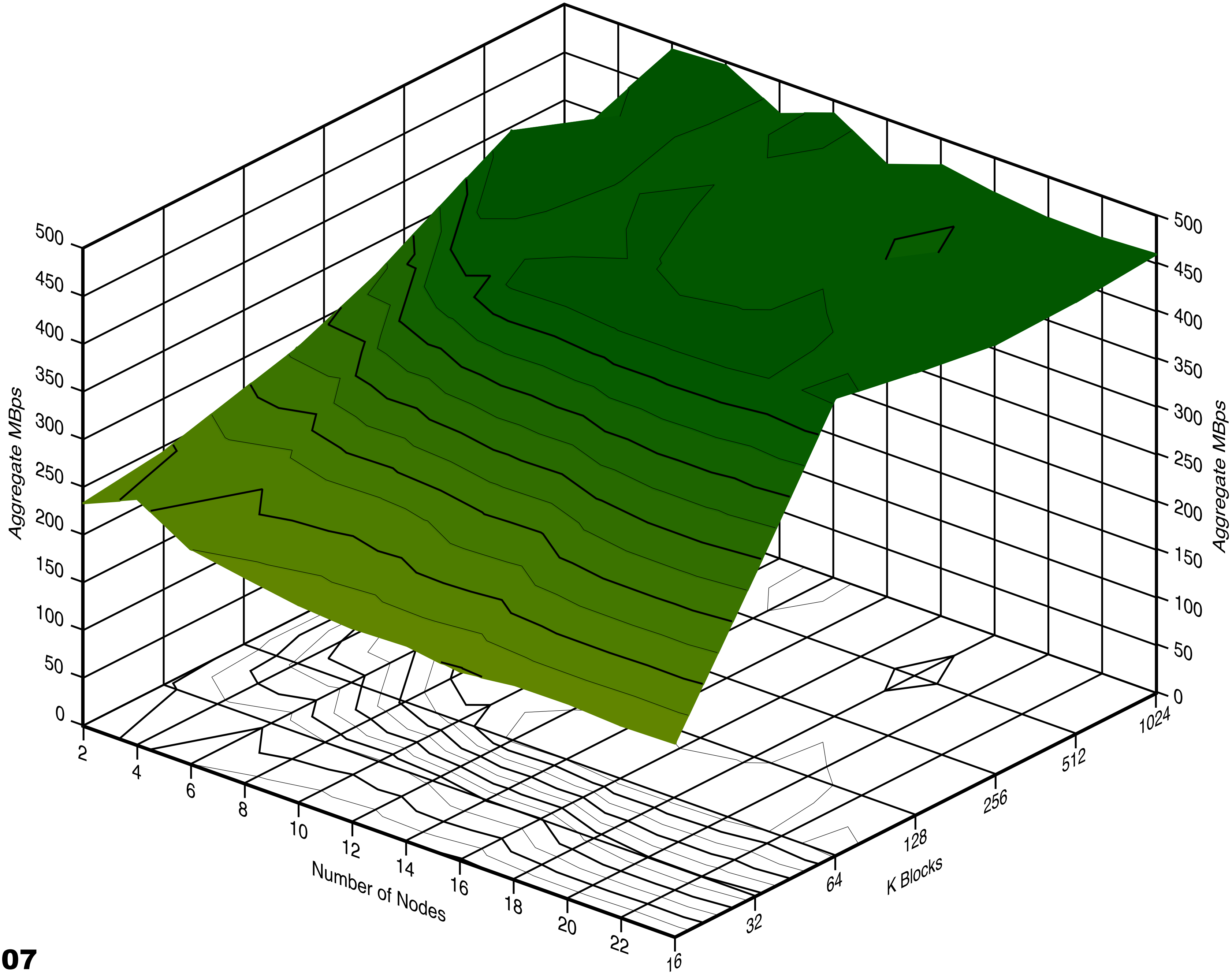
Sample Results: Uniformity

~10% Variation across a 12.5 hour run. [24 : 1 : * : 1,0 : 0,0] POS:CR:64K:310M - SLES9 2.6.5-7.244 with 6x 802.3ad



Sample Results: Scalability

[VAR : 1 : 1 : 1,0 : 0,0] POS:RW:VAR:500M - SLES10 2.6.16.21-0.8 with 6x Dedicated @ 0% Spatial Utilization



Some Possible Future Directions for Bringsel

- Code refinement, documentation.

- Tree discovery/tree limit.

- UPC support.

- Adding and pruning directories in CF.

- Selectable horiz/vert barriers.

- Fault injection.

- Parser refinements.

- Modules to support tracing output, either VFS or library level.

- Better visualization methods (external).

- Long term, automated style driver (external).

Questions?