Ottawa Linux Symposium 2007 Discussion

Filesystem Support for Continuous Snapshotting

Ryusuke Konishi, Koji Sato, Yoshiji Amagai

{ryusuke,koji,amagai}@osrg.net

NILFS Team, NTT Labs.

Agenda

- **What is Continuous Snapshotting?**
- **Continuous Snapshotting Demo**
- **Brief overview of NILFS filesystem**
- **Performance**
- **Kernel issues**
- **Free Discussion**
  - Applications, other approaches, and so on.

What's Continuous Snapshotting?

- **Technique creating number of checkpoints (recovery points) continuously.**
  - can restore files mistakenly overwritten or destroyed even the mis-operation happened a few seconds ago.
  - need no explicit instruction **BEFORE** (unexpected) data loss
  - Instantaneous and automatic creation
  - No inconvenient limits on the number of recovery points.

What's Continuous Snapshotting?

- **Typical backup techniques...**
  - make a few recovery points a day
  - have a limit on number of recovery points
    - e.g. The Vista Shadow Copy (VSS):

      1 snapshot a day (by default), up to 64 snapshots, not instant.

User's merits

- **Receive full benefit of snapshotting**
  - **For general desktop users**
    - No need to append versions to filename; document folders become cleaner.
    - can take the plunge and **delete** (or overwrite save).
    - Possible application to regulatory compliance (i.e. SOX act).
  - **For system administrators and operators**
    - can help online backup, online data restoration.
    - allow rollback to past system states and safer system upgrade.
    - Tamper detection or recovery of contaminated hard drives.

Continuous Snapshotting Demo (NILFS)

- **A realization of Continuous Snapshotting shown through a Browser Interface**
- **Online Disk Space Reclaiming (NILFSv2)**

NILFS project

- **NILFS(v1) released in Sept, 2005.**
  - The first version which lacks GC (Cleaner).
- **NILFS2(v2) released in June, 2007.**
  - Supports online GC with maintaining multiple snapshots.
  - Supports kernel 2.6.11~2.6.21
- **NILFS project home page**
  - **http://www.nilfs.org/**
  - **GPL software; downloadable from the site.**
  - NILFS Mailing List (in English)

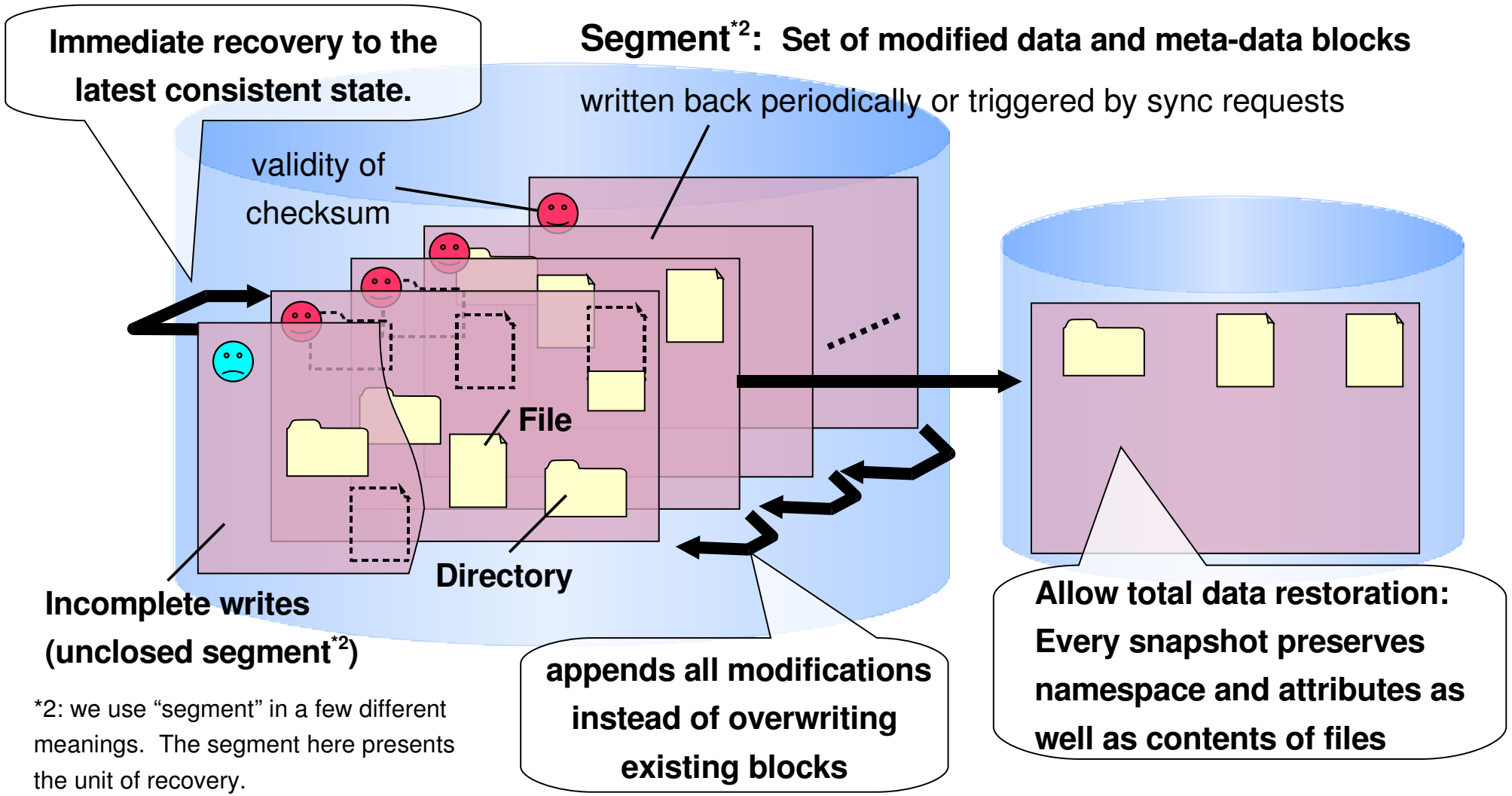
Filesystems with Snapshots

Filesystem	Developer	Max. Number of Snapshots	Instant Snapshotting	Sustainable Snapshots after GC	Can change past checkpoint into snapshot
ZFS	Sun Microsystems	No limits		–	
NTFS	Microsoft	64		–	
Ext3cow	Johns Hopkins University	No limits		–	
LFS	UCB	0		0	
Linux LFS with GC	Charles University (Prague)	No limits	√	1	
NILFS2	NTT	No limits	√	No limits	√

Features of NILFS

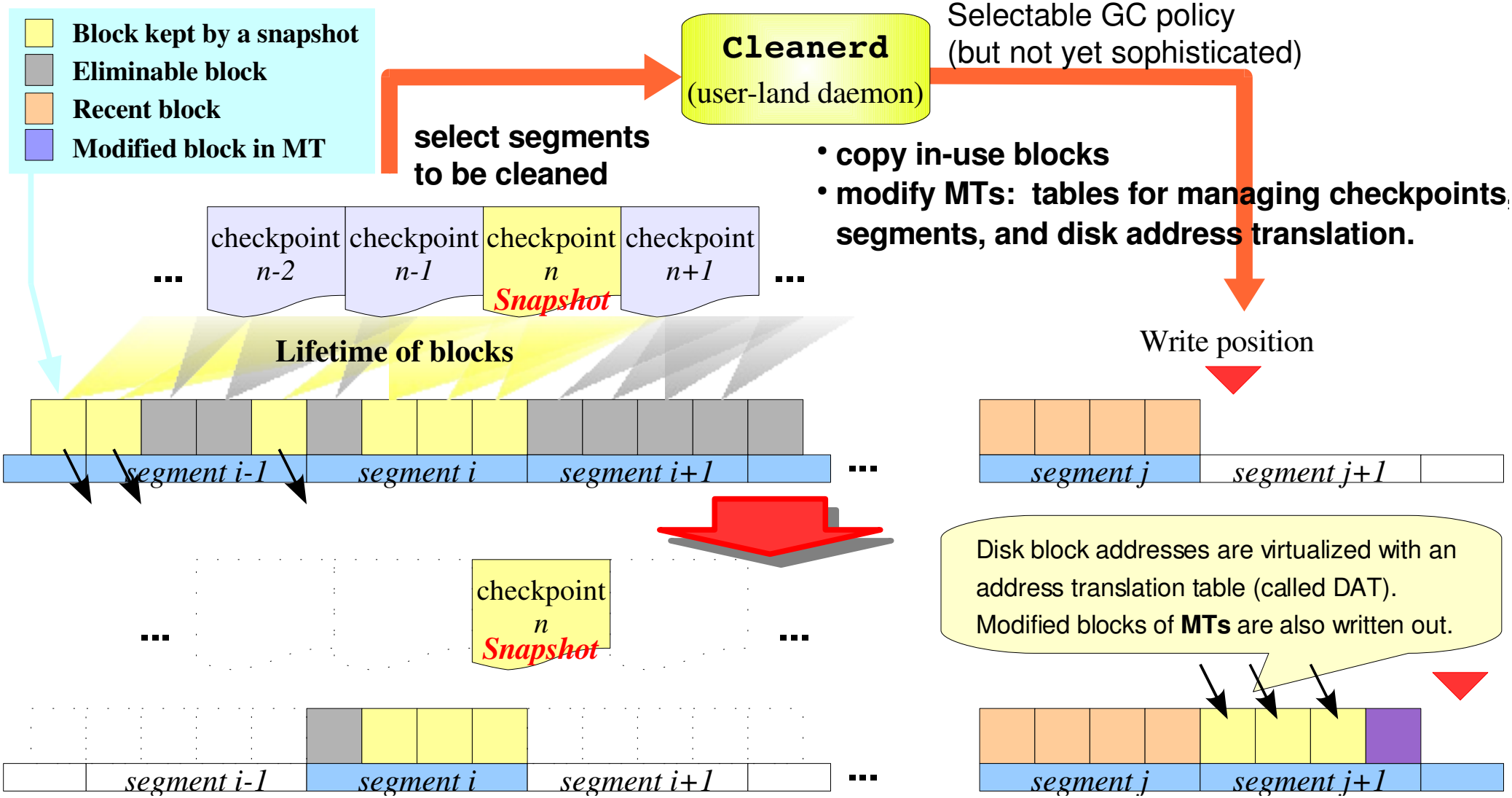
- **Continuous snapshots**
  - Every snapshot can be accessed as a normal RO file-system
- **Practical Log-structured Filesystem for Linux**
  - **B-tree based block and meta-data management**
  - **64-bit data structures**
    - support many files, large files and disks.
  - **Immediate recovery after system crash**
    - Highly available like journaling filesystems
  - **Loadable Kernel module (No kernel patch required)**

Conceptual diagram of NILFS snapshotting



*2: we use “segment” in a few different meanings. The segment here presents the unit of recovery.

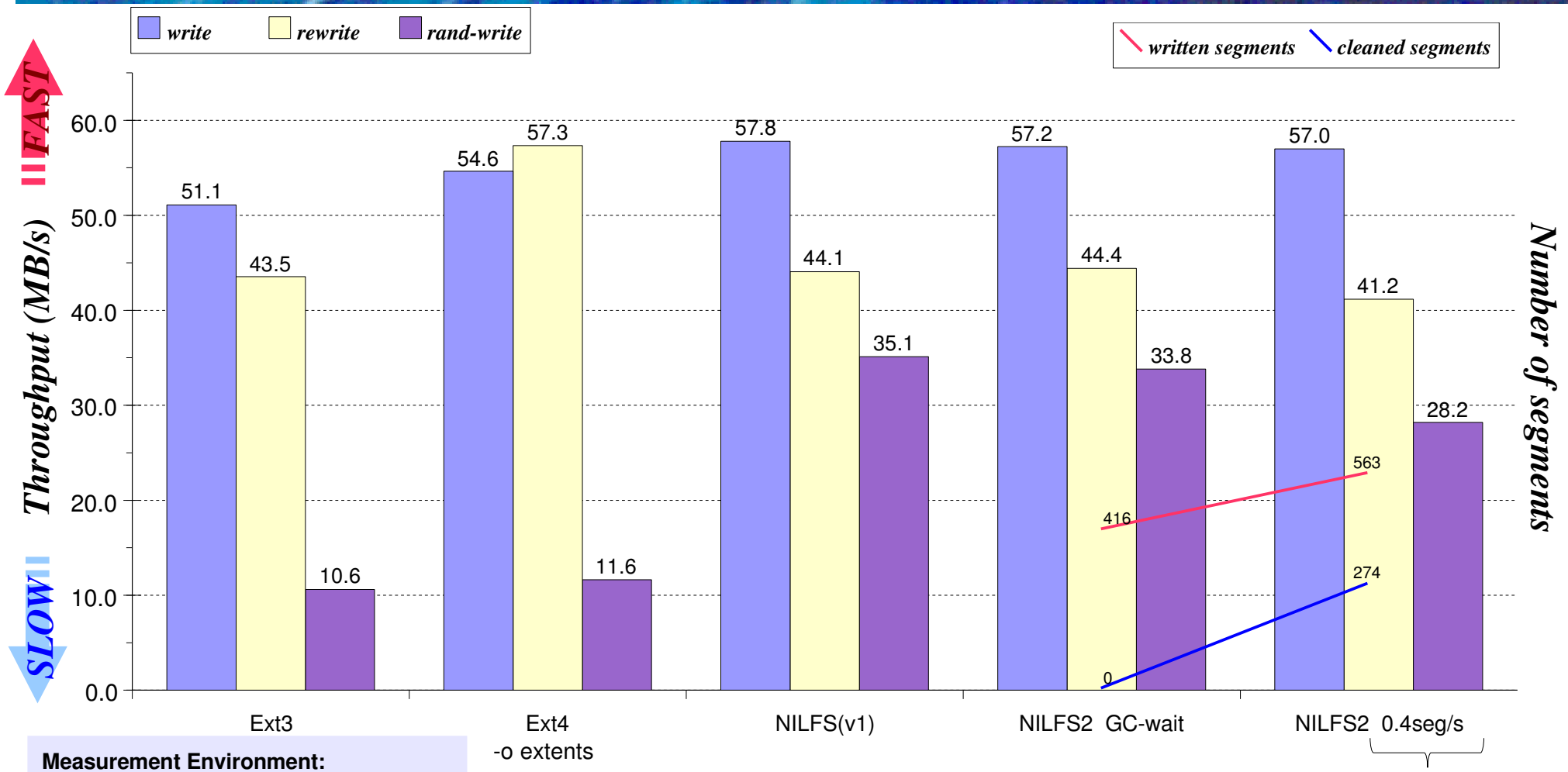
Online Disk Space Reclaiming (NILFS2)



(1) Performance

- **Discussion Point**
  - Though LFS approach shows high write performance, it incurs performance penalty due to fragmentation and GC overhead. (shown in the following slides)
- **Question**
  - Acceptable in return for the feature?

(1) Performance – iozone write

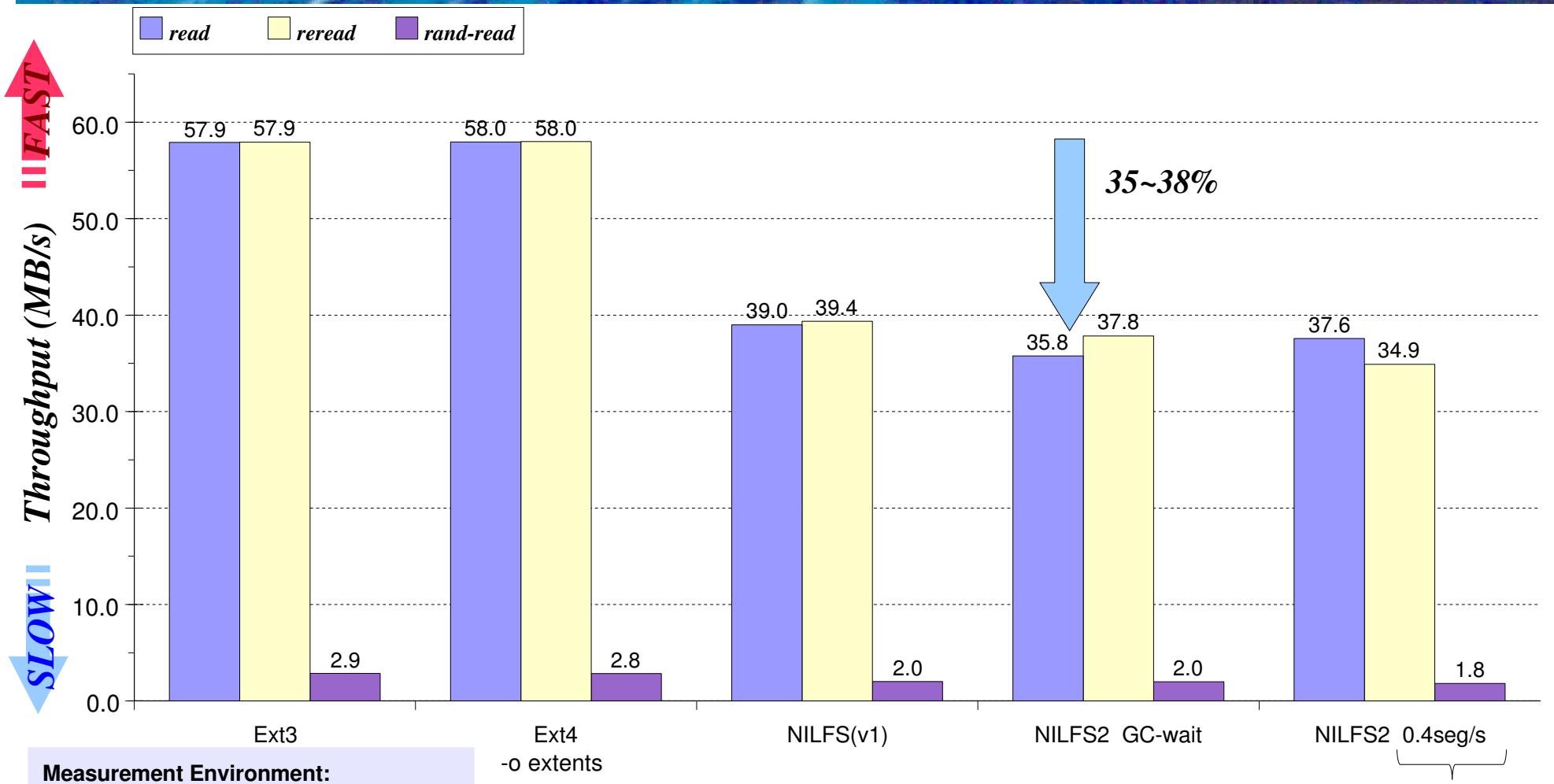


Measurement Environment:
- Linux 2.6.20 SMP PREEMPT x86_64
- EM64T Pentium 4 HT 3.2GHz / 2GB RAM
- 80GB SATA drive x2 (sda: system, sdb: test)

Reclamation Speed: number of segments cleaned per second (1segment = 8MB)

```
iozone -i 0 -i 1 -i 2 -s 1g -r 16 -e -U /test -f /test/testcase
```

(1) Performance – iozone read



Measurement Environment:
- Linux 2.6.20 SMP PREEMPT x86_64
- EM64T Pentium 4 HT 3.2GHz / 2GB RAM
- 80GB SATA drive x2 (sda: system, sdb: test)

Reclamation Speed: number of segments cleaned per second (1segment = 8MB)

```
iozone -i 0 -i 1 -i 2 -s 1g -r 16 -e -U /test -f /test/testcase
```

(2) Page Cache for Continuous Snapshotting

- **Discussion Point**
  - Current NILFS design avoids applying no specialized versioning extension to page cache itself.
    - Every snapshot has independent page caches.
    - Concurrent access to different snapshots is not efficient both in memory utilization and read performance.
- **Question**
  - What kind of page cache enhancement is reasonable and acceptable for continuous snapshotting?

(3) Online Block Relocation

- **Discussion Point**
  - LFS changes on-disk address of data and meta-data in each write or when GC moves them to reclaim disk space.
    - To avoid filesystem failure or unsecured data read due to reuse of disk blocks, the start and end of disk read must be recognizable.
    - But FS cannot know the completion of some type of read.
      e.g. readv(), io_submit()
    - Mapped flag (BH_Mapped) interferes reassignment of disk address.
- **Question**
  - Better kernel support to achieve safe block relocation.
    - It seems a common issue on operations like the online defrag.

Free Discussion

- **How is Continuous Snapshotting?**
- **Other approaches to Continuous Snapshotting**
- **Applications**
- **Garbage Collection Strategy**
- **How to present snapshots?**
  - Mount point per snapshot, or extended namespace