Device removal

Matt Ahrens (mahrens@delphix.com)
Alex Reece (alex@delphix.com)

Why?

- Customers over provision
- “Oops, that was supposed to be a mirror”

- (only top level devices)

How?

- Disable allocation to that device
- Copy all data to other devices

- Where did the data go?

Keeping track of data (bad) ...

- Traverse block pointers via scan
- Map { *old BP -> new BP* }
- Lookup BP on read, free (repair?)
- On disk and in memory

Keeping track of data (bad) ...

- Traverse block pointers via scan
- Map { *old BP -> new BP* }
- Lookup BP on read, free (repair?)
- On disk and in memory
  - Huge table!

Keeping track of data (bad) ...

- Traverse block pointers via scan
- Map { *old BP -> new BP* }
- Lookup BP on read, free (repair?)
- On disk and in memory
  - Huge table!
  - BP rewrite?

Keeping track of data (good) ...

- Traverse allocated segments on disk
- Map *{ old segment -> new segment }*

Keeping track of data (good) ...

- Traverse allocated segments on disk
- Map { *old segment -> new segment* }
- Lookup BP on read, free (repair?)
- On disk and in memory

Keeping track of data (good) ...

- Traverse allocated segments on disk
- Map *{ old offset, length -> device, new offset }*
- Lookup BP on read, free (repair?)
- On disk and in memory

Keeping track of data (good) ...

- Traverse allocated segments on disk
- Map *{ old offset, length -> device, new offset }*
- Lookup BP on read, free (repair?)
- On disk and in memory
    - Save space

Keeping track of data (good) ...

- Traverse allocated segments on disk
  - LBA order
- Map *{ old offset, length -> device, new offset }*
- Lookup BP on read, free (repair?)
- On disk and in memory
  - Save space

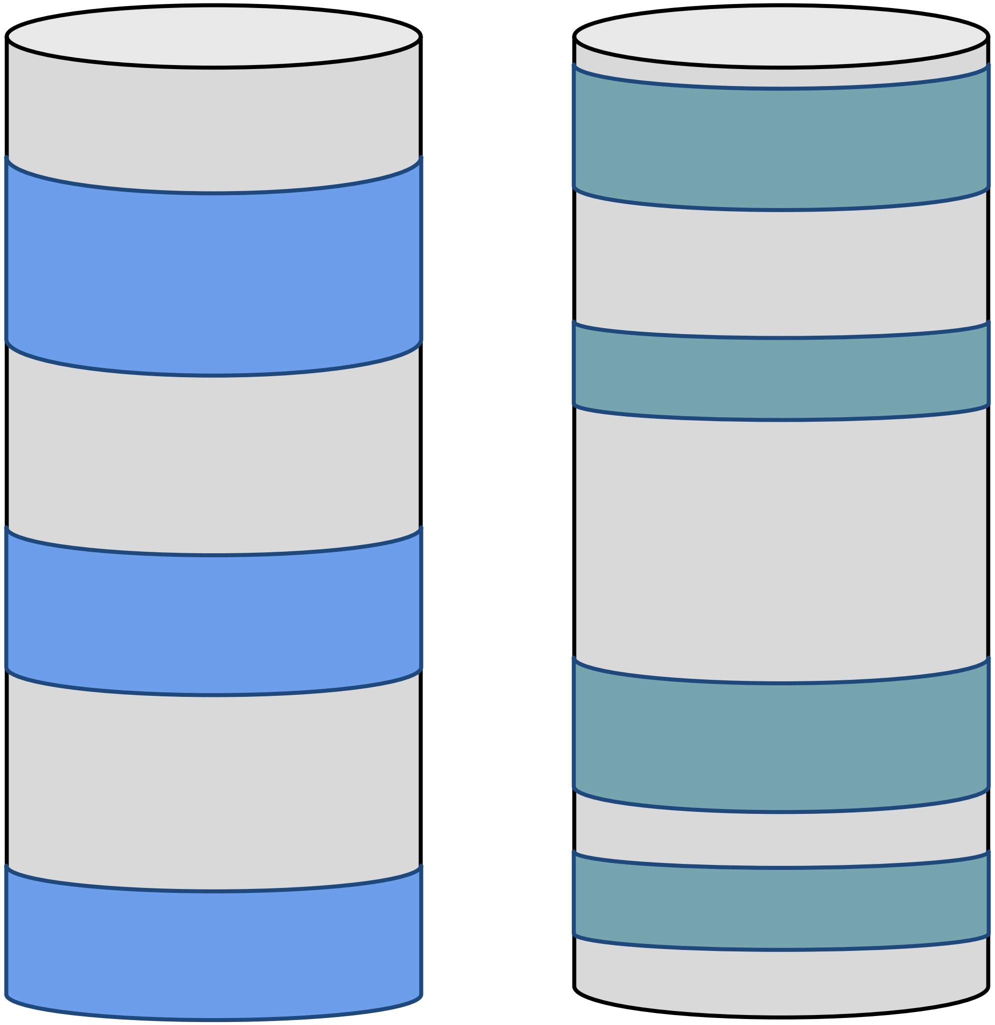
Not so fast!

- Removal implementation details
    - Minimize effect on system
    - Understandable workflow
    - Work with other ZFS features
- Post removal performance
    - Memory overhead

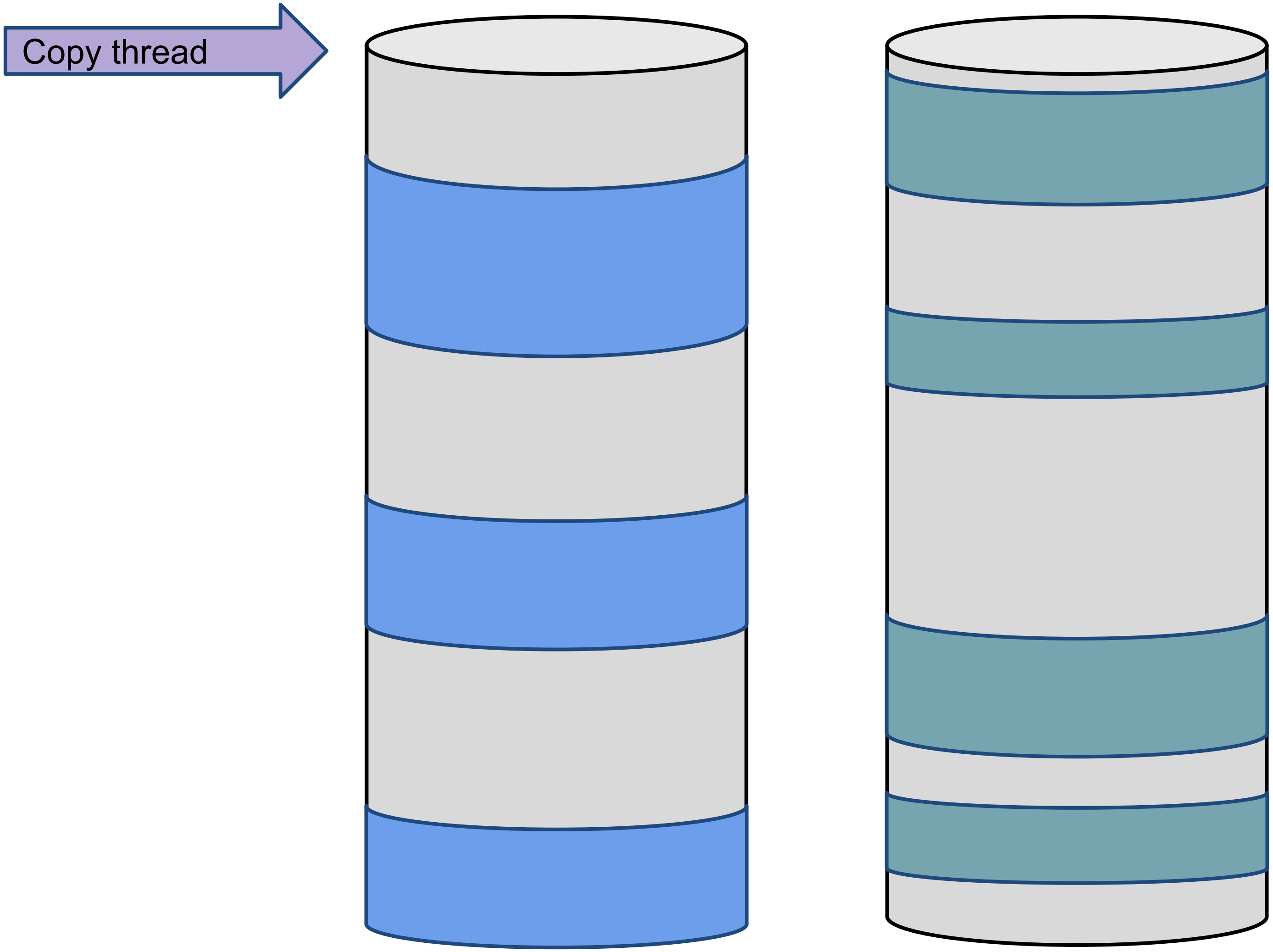
Minimize effect on system.

- How did scrub work?
    - Time slice between write and scrubs
- Open context removal
    - Copy thread issue ios to copy
    - Copy thread updates sync thread
    - Sync thread updates partial mapping

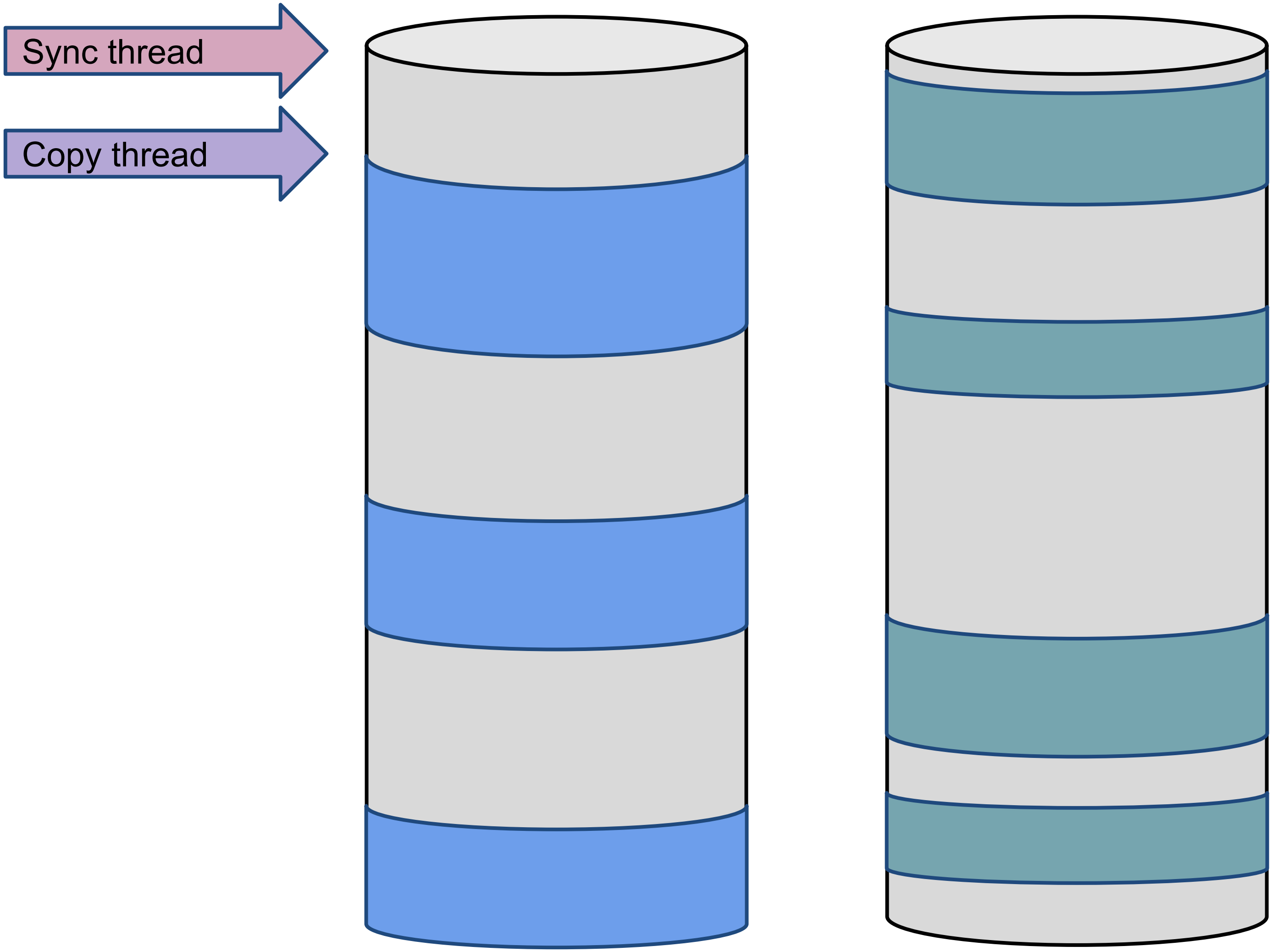
Open context removal



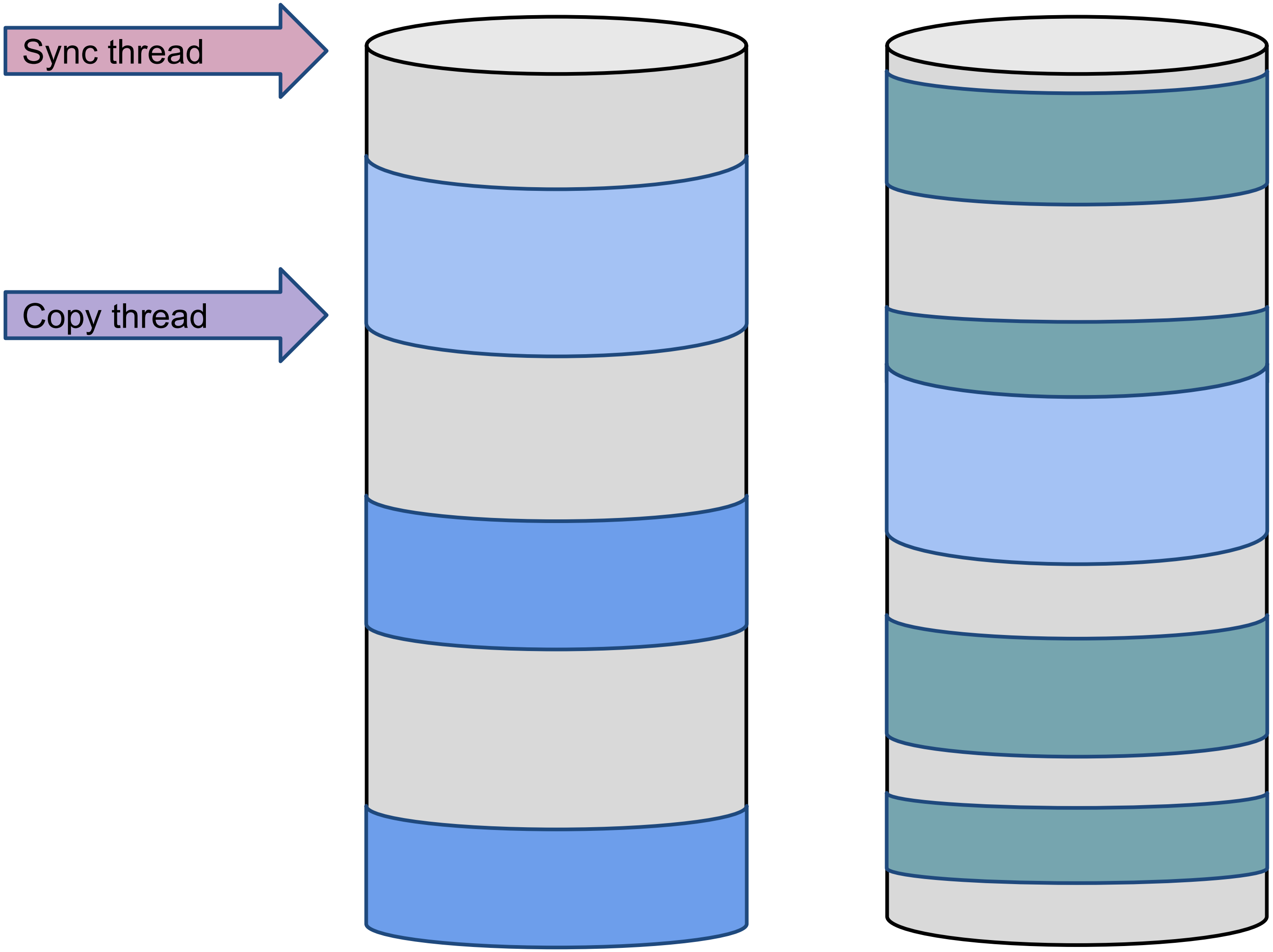
Open context removal



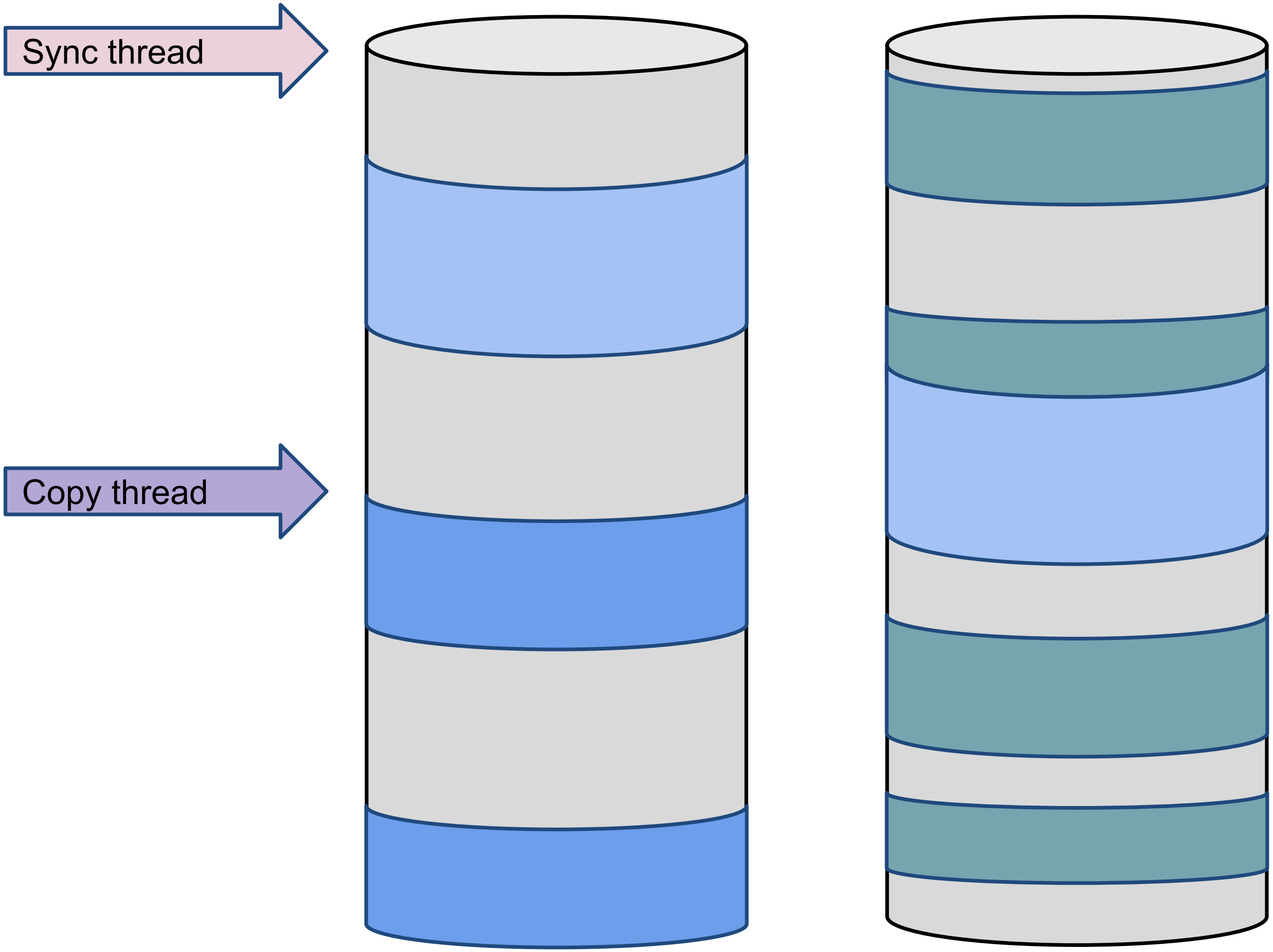
Open context removal



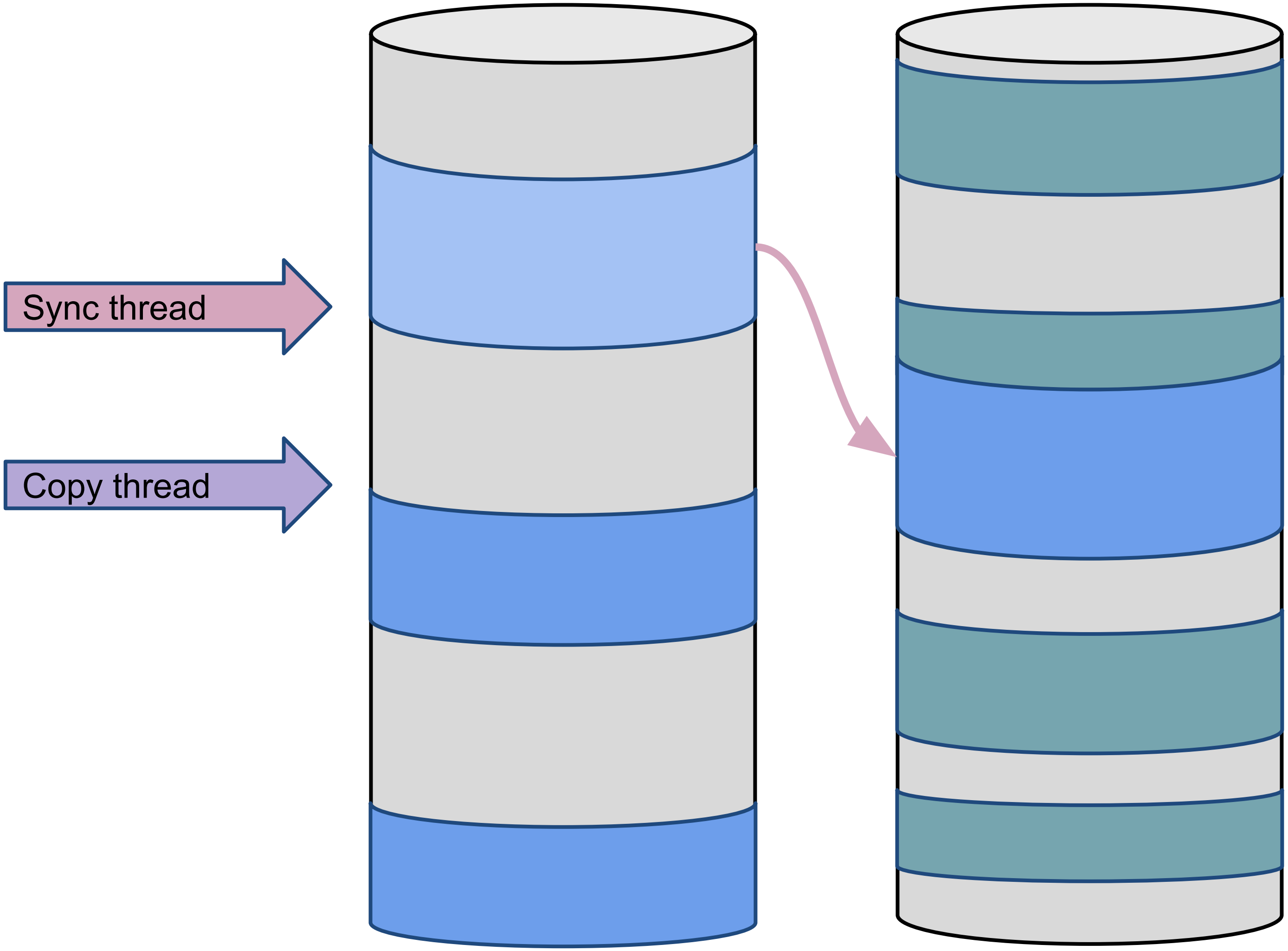
Open context removal



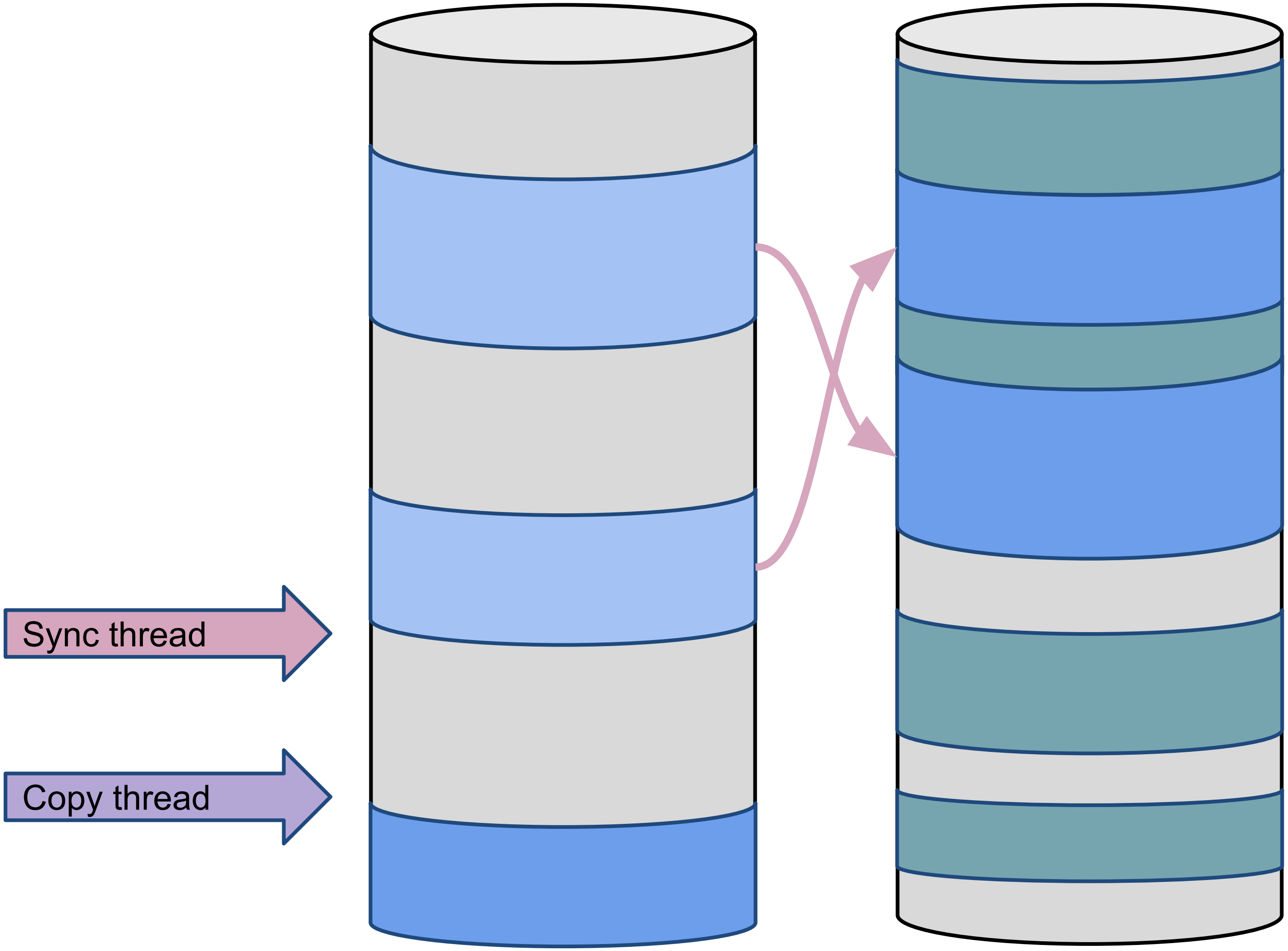
Open context removal



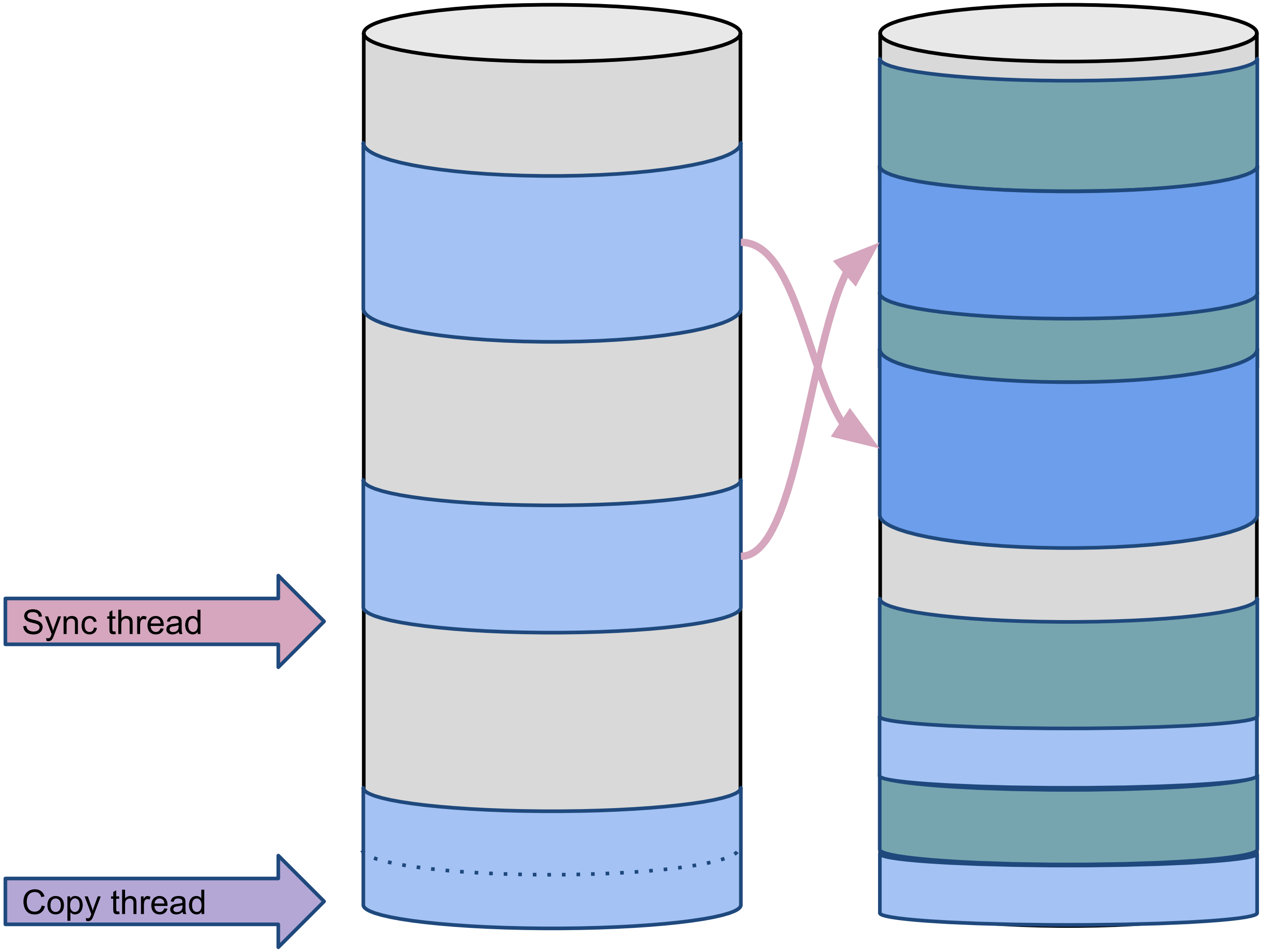
Open context removal



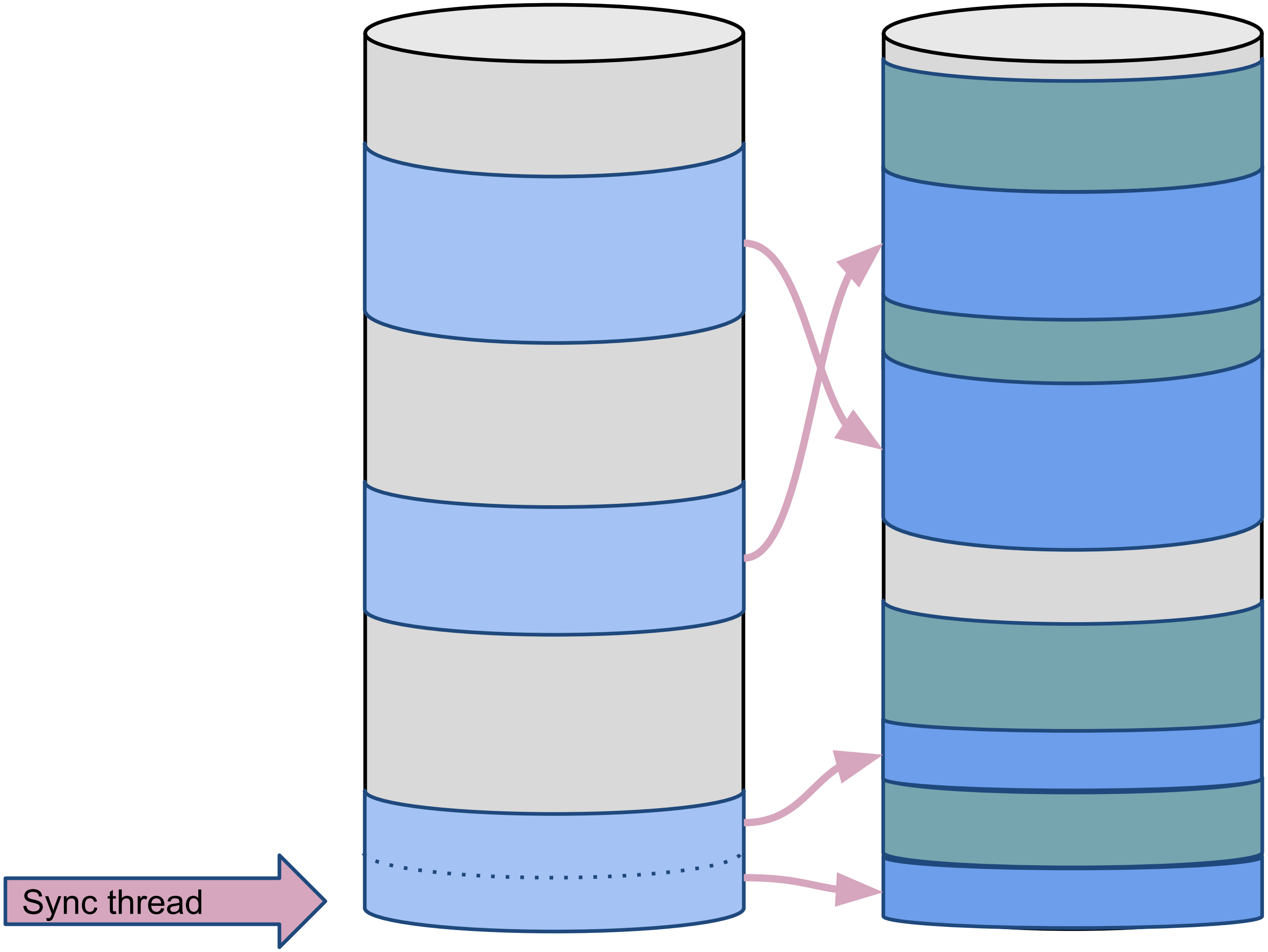
Open context removal



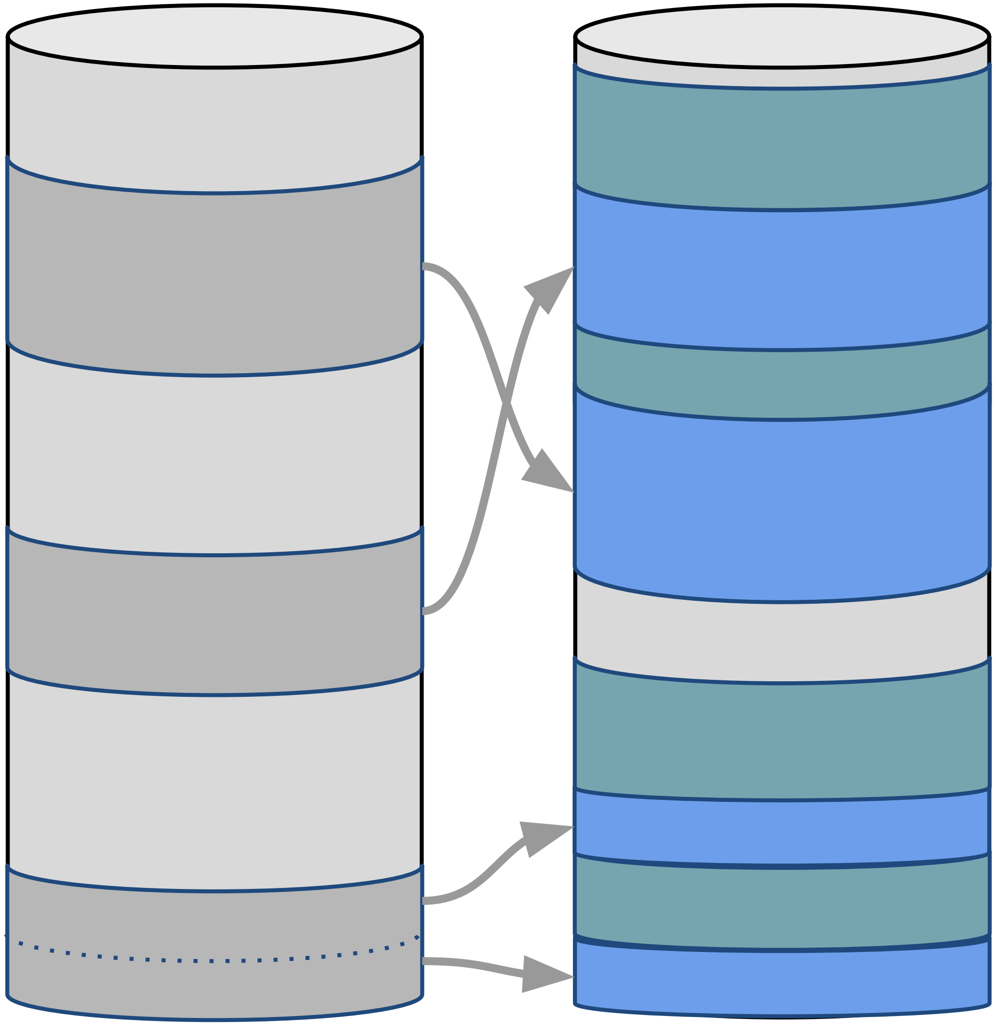
Open context removal



Open context removal



Open context removal



Open context removal

- Scan space maps
- Mapping entry covers allocated segment
- Split large allocated segments
- Cannot modify space maps during removal

Deferred frees

- Potentially a lot of unfreeable space
- Defer only in flight frees?

How to make workflow clear?

- Predicting memory usage
- Space accounting
- Progress reporting
- Cancellation

Other ZFS features

- dedup, compression, snapshots, clones - works
- checksums - doesn’t verify checksums
- scrub/resilver - works
- RAID-Z - kinda works

Perf issues after removal

- indirection lookup (probably minimal)
- memory overhead (substantial; must mitigate)
    - ~1GB per 1TB of data
    - Map fragmented regions
    - Rewrite BPs for active filesystems
    - Evict unused parts of mapping
    - Garbage collection

Status

- ETA to upstream
- Demo!

Demo

```
$ sudo zpool remove test c2t2d0

$ sudo zpool status -v test

  pool: test

 state: ONLINE

  scan: none requested

remove: Evacuation of vdev 1 in progress since Mon Nov 10 08:06:43 2014

    340M copied out of 405M at 67.5M/s,  83.90% done, 0h0m to go

config:

        NAME        STATE     READ WRITE CKSUM

        test        ONLINE       0     0     0

          c2t1d0    ONLINE       0     0     0

          c2t2d0    ONLINE       0     0     0

          c2t3d0    ONLINE       0     0     0
```