AIO in Postgres 18 and beyond

Andres Freund
PostgreSQL Developer & Committer
Email: andres@anarazel.de
Email: andres.freund@microsoft.com

https://anarazel.de/talks/2025-09-30-pgconf-nyc-aio-in-PG-18-and-beyond/aio-in-PG-18-and-beyond.pdf

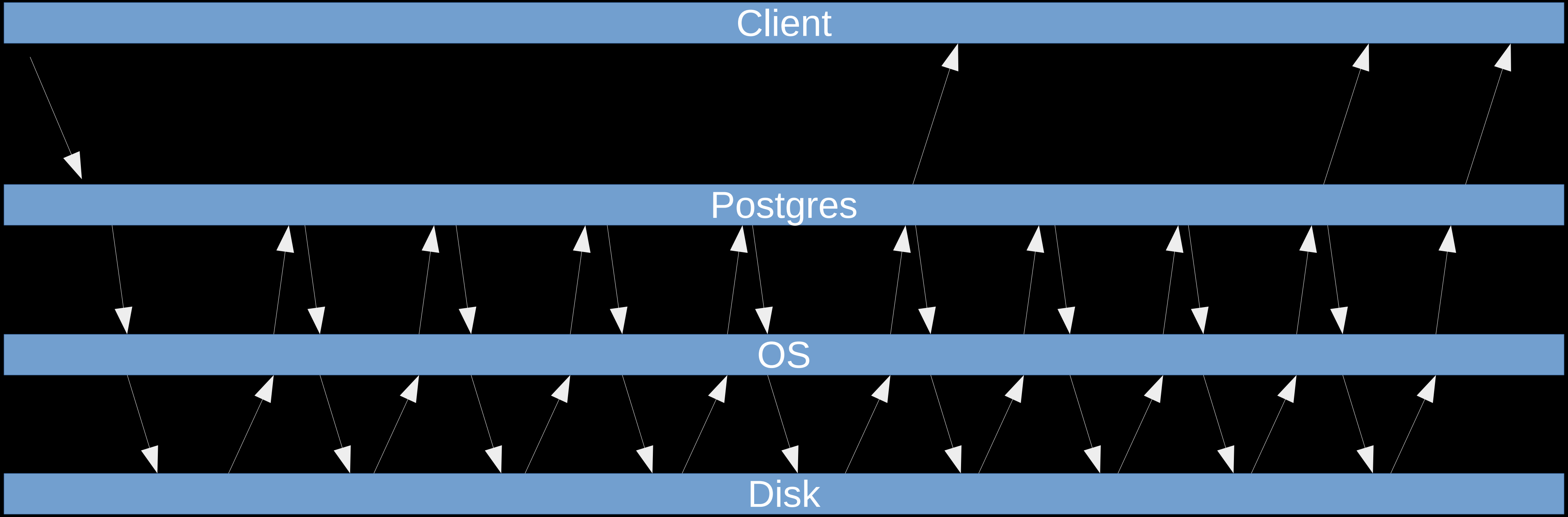
Thanks

- A lot of work by a lot of folks
- Thomas, Melanie, Bilal, Noah, Heikki, Robert, …
- Microsoft
- Tomas and Peter G. for index prefetching work
- Lots of others

AIO?

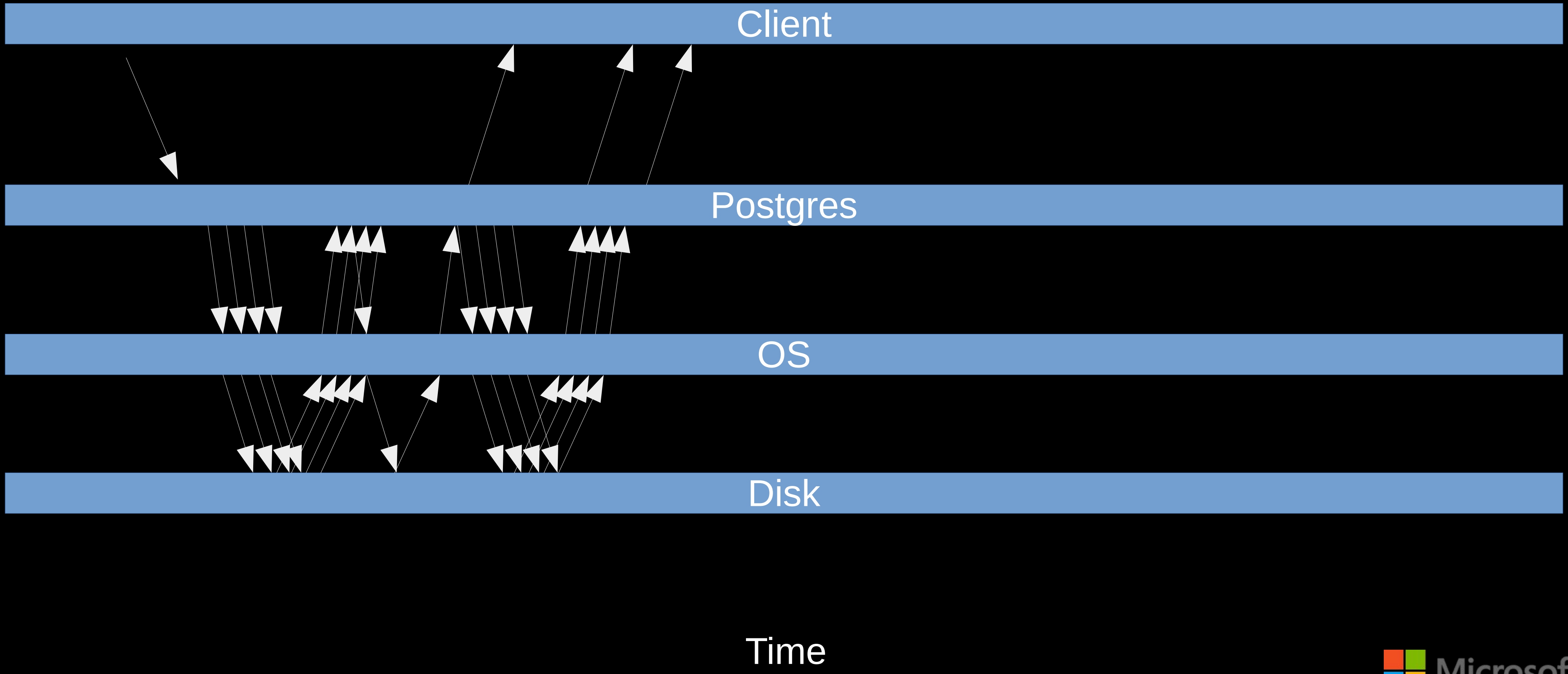
Asynchronous Input/Output

synchronous, not cached



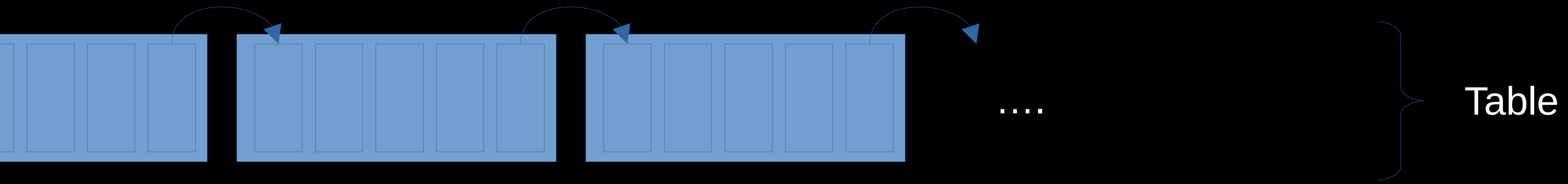
Time

asynchronous, not cached

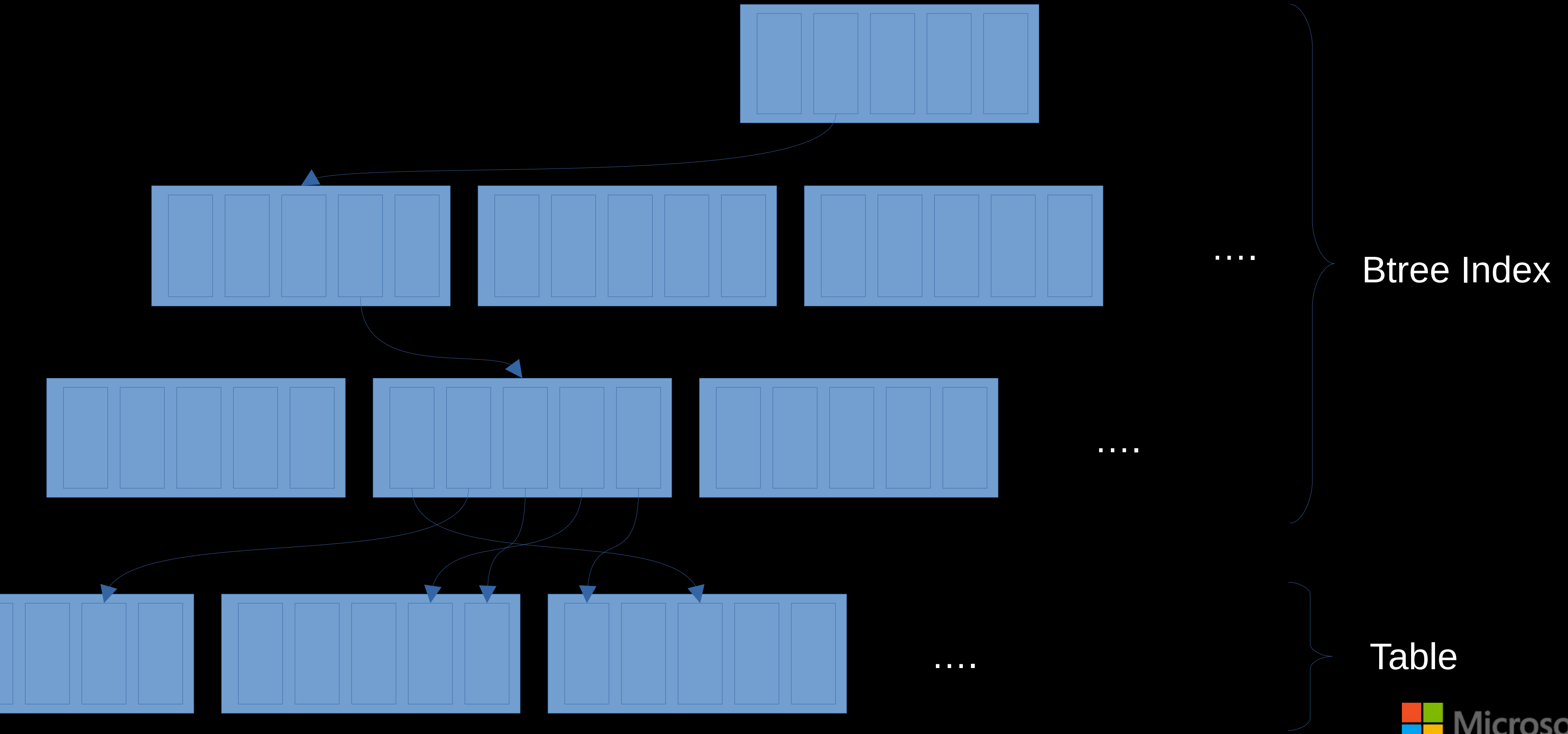


Predict the Future

Sequential Scan



Index Scan



16:

- Buffer Mangager Infra
- Relation Extension

17:

- Read Streams
- Streamify
    - Seq Scan
    - Analyze
    - Prewarm
- Experimental Direct I/O

18:

- AIO Infra
- AIO for buffered reads
- Streamify
    - Bitmap Heap Scan
    - Vacuum
    - autoprewarm
    - CREATE DATABASE
    - amcheck

18: io_method = worker

- portable
- parallelizes checksums, memory copy
- limited I/O depth, particularly with high latency storage
- global
- number of workers controlled by io_workers

18: io_method = io_uring

- linux specific, better with recent-ish kernels
- lower latency
- deep I/O queues
- per backend
- does **not** parallelize checksum computation
- requires tuning of file descriptor limits

18: io_method = sync

- don’t use AIO
- behaves as close as realistic to < 18
- “safety net”

18: When can AIO help?

- IO bound
  - track_io_timing
  - EXPLAIN (ANALYZE, BUFFERS)
- only for reads
- foreground: seqscan, bitmap heap scan
- background: vacuum
- Just the absolute basics!

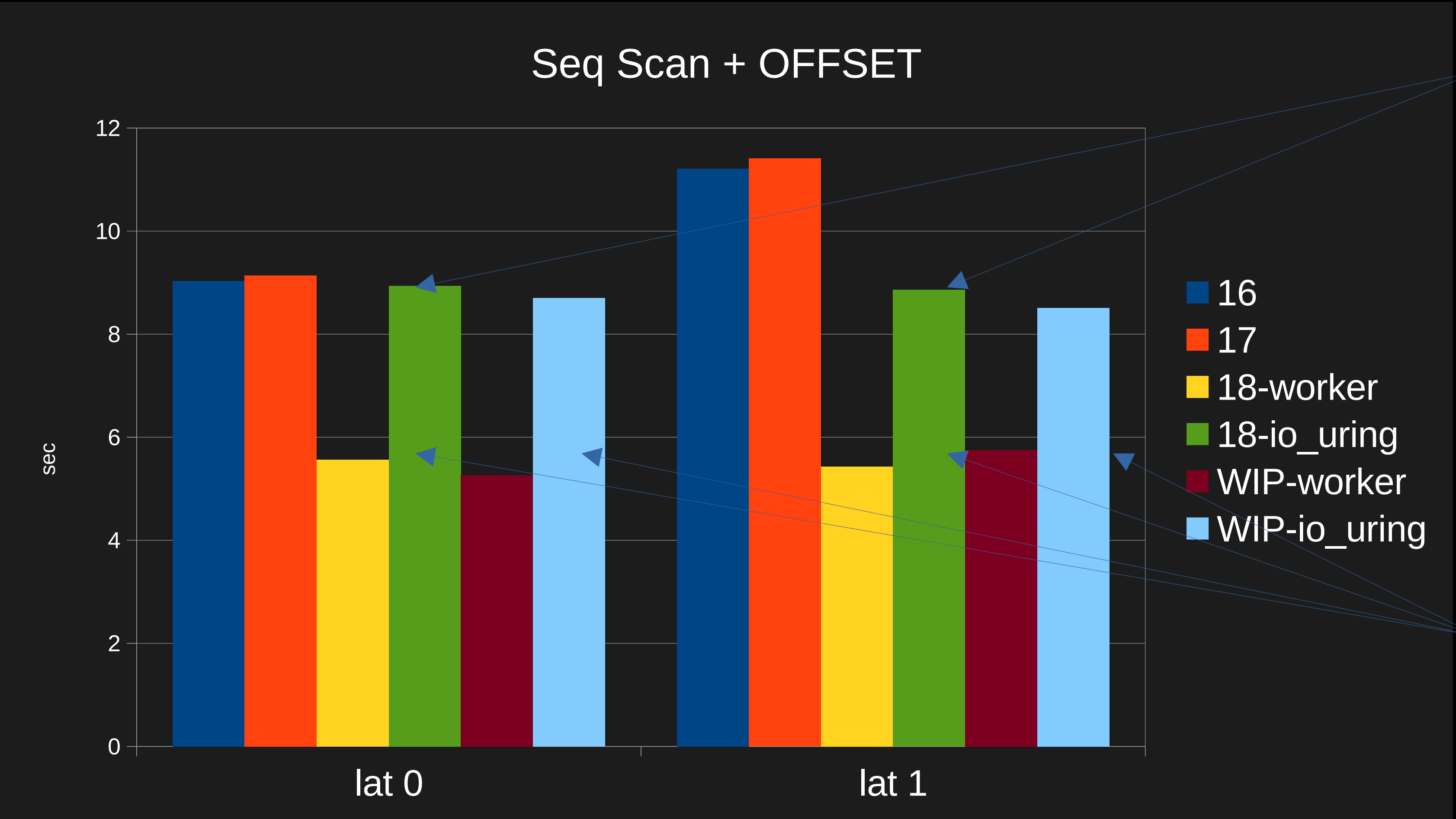
Benchmark Setup

- 2x Gold 6442Y, 256GB RAM
- 2x Samsung SSD 2TB PM9A1, striped, XFS
- Linux 6.17
- Artificial 1ms latency added with dm_delay
- io_workers=32, effective_io_concurrency=32, shared_buffers=32GB
- checksums enabled on all branches
- 16 has support for cache clearing added

Benchmark Workload

- Large table with sequential and random columns
- Table populated in parallel
- PG & OS cache is cleared between queries

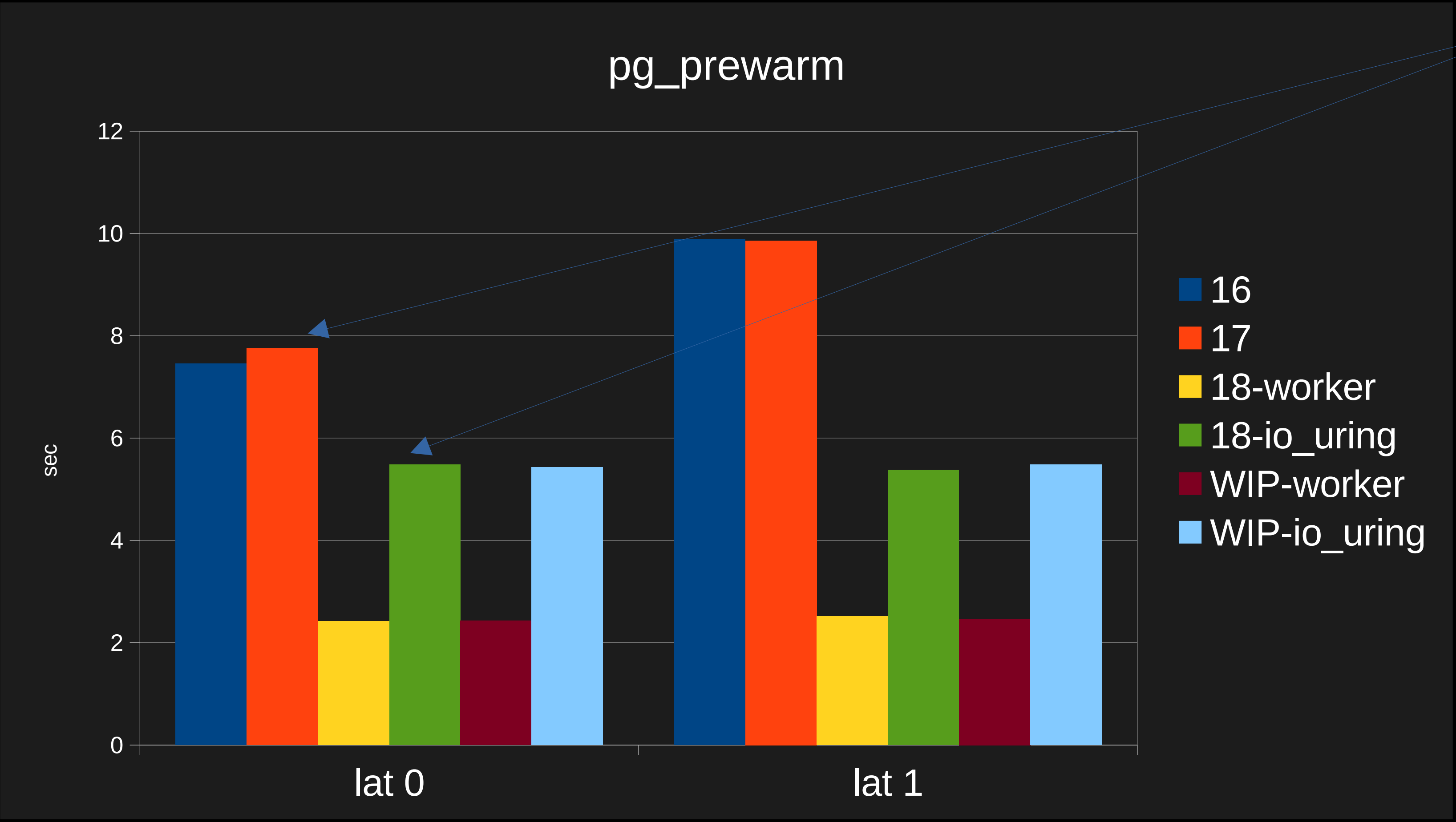
18: AIO for Seq Scans



- Latency has little effect with AIO
- CPU bottlenecked (query + checksums)

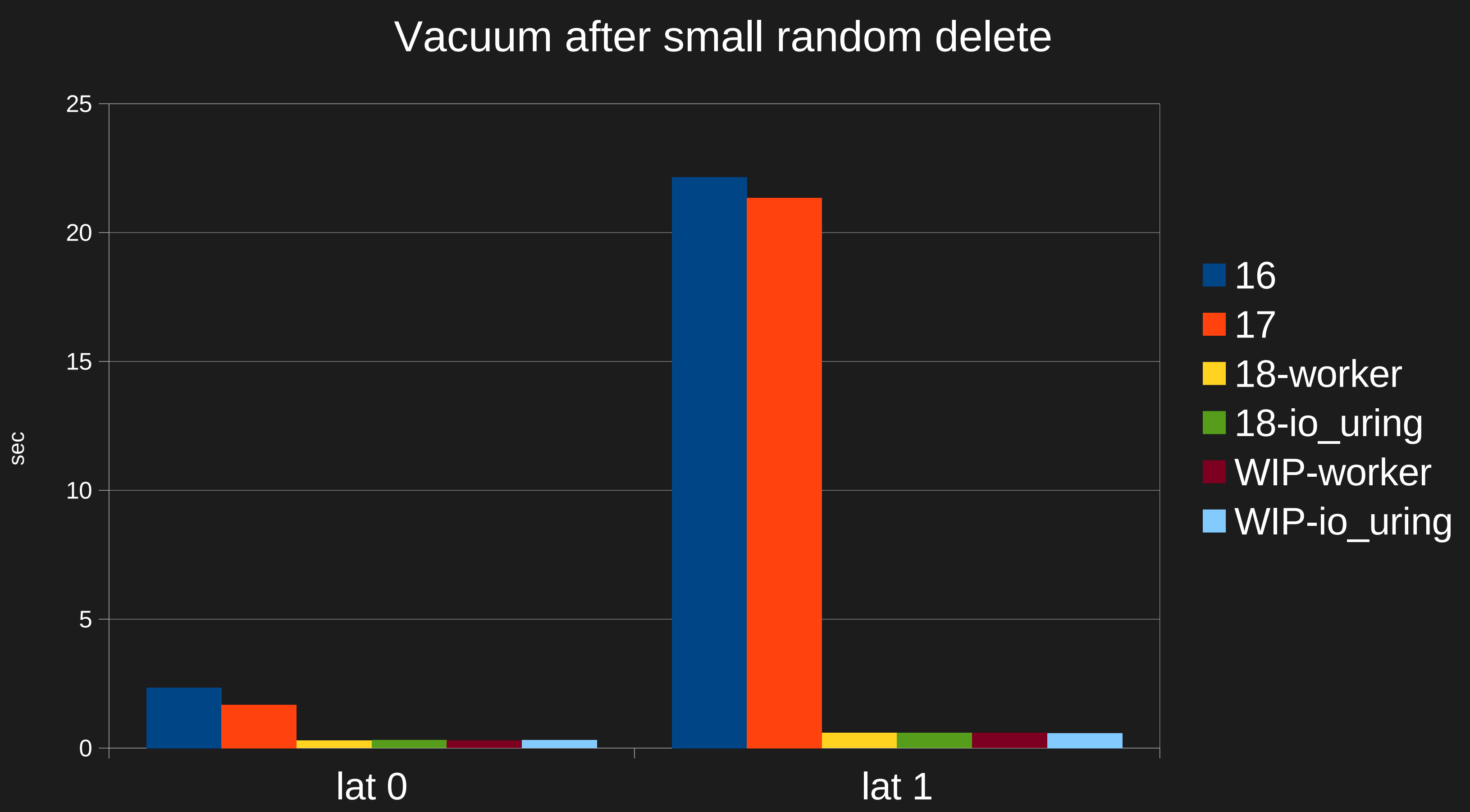
Checksums not parallelized

18: AIO for prewarm



- bigger difference without checksum

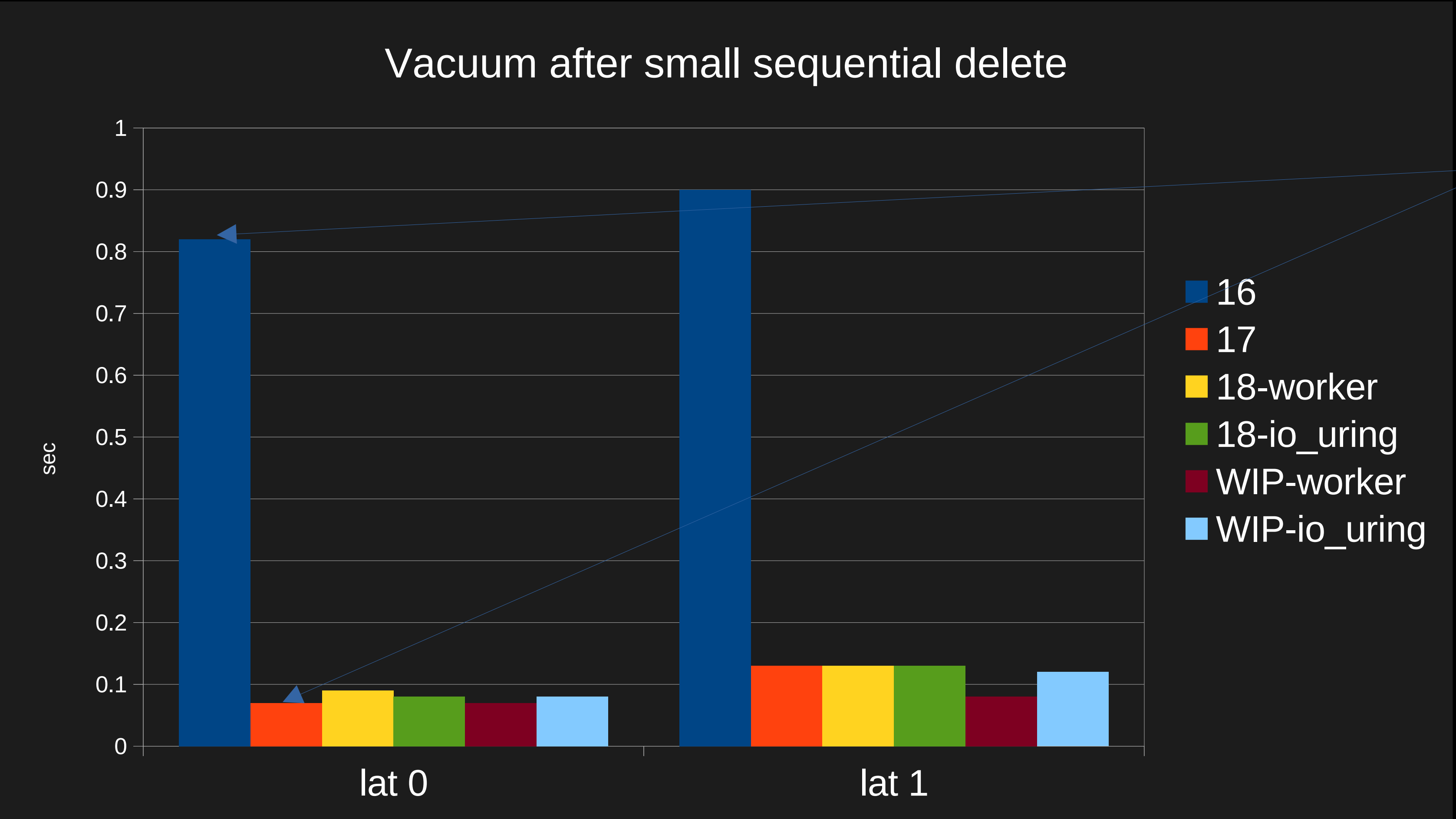
18: AIO for Vacuum



Vacuum after small random delete

- huge effect
- only if small portion of table changed
  - can be generalized

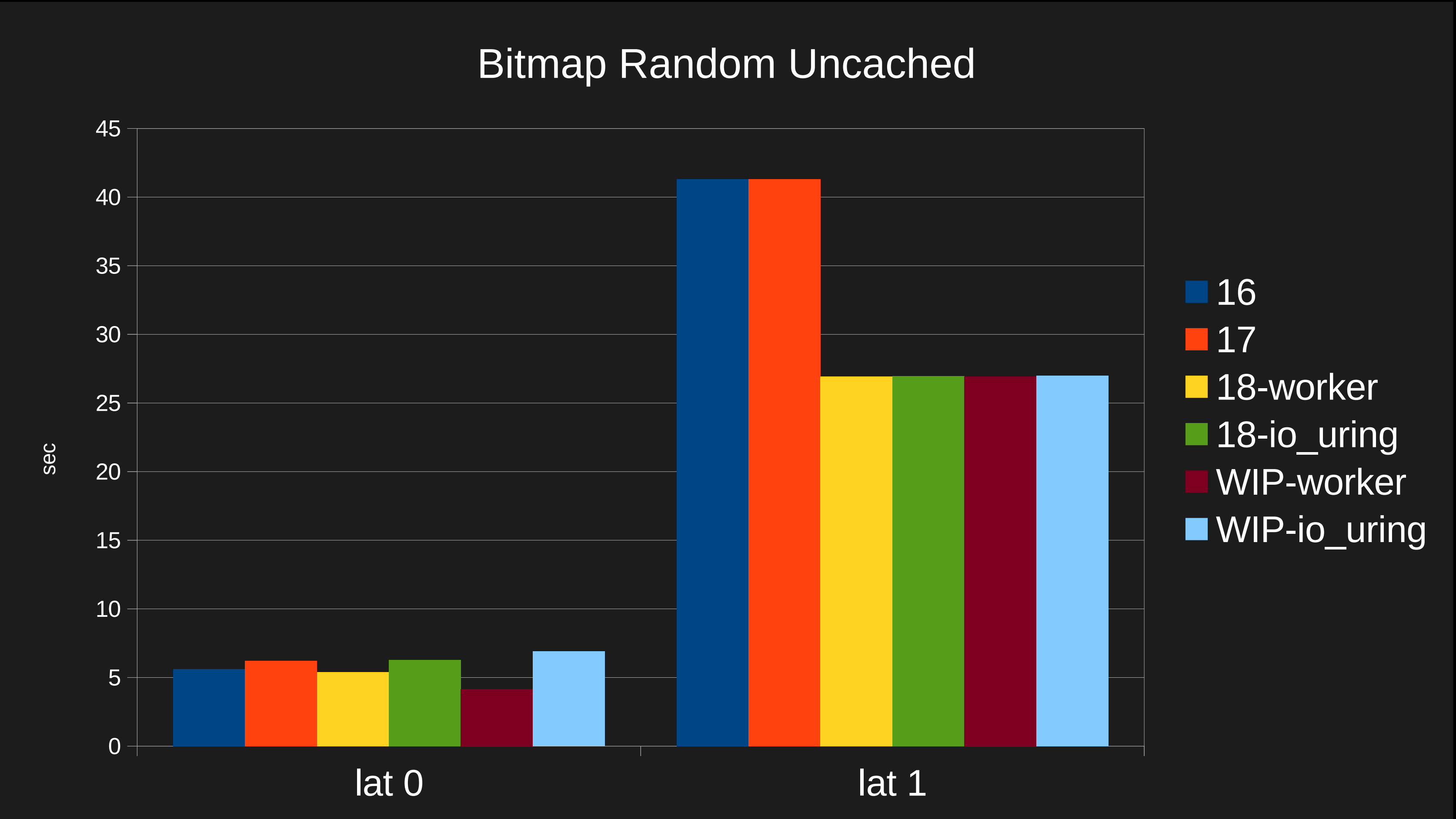
18: AIO for Vacuum



- AIO has no effect → OS readahead
- CPU efficiency improvement in 17

Microsoft

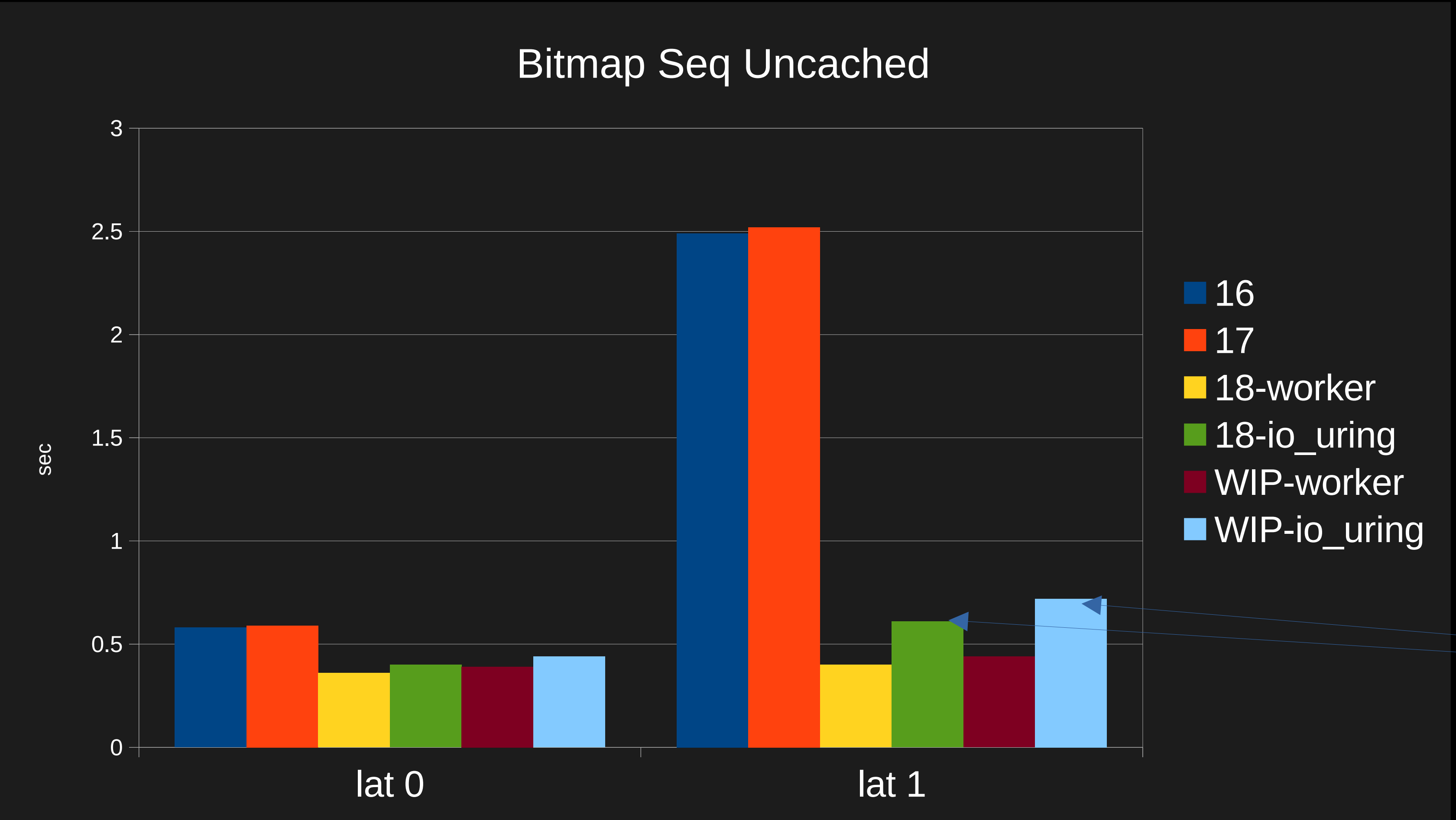
18: AIO for Bitmap Heap Scan



- already did prefetching
- but now with DIO

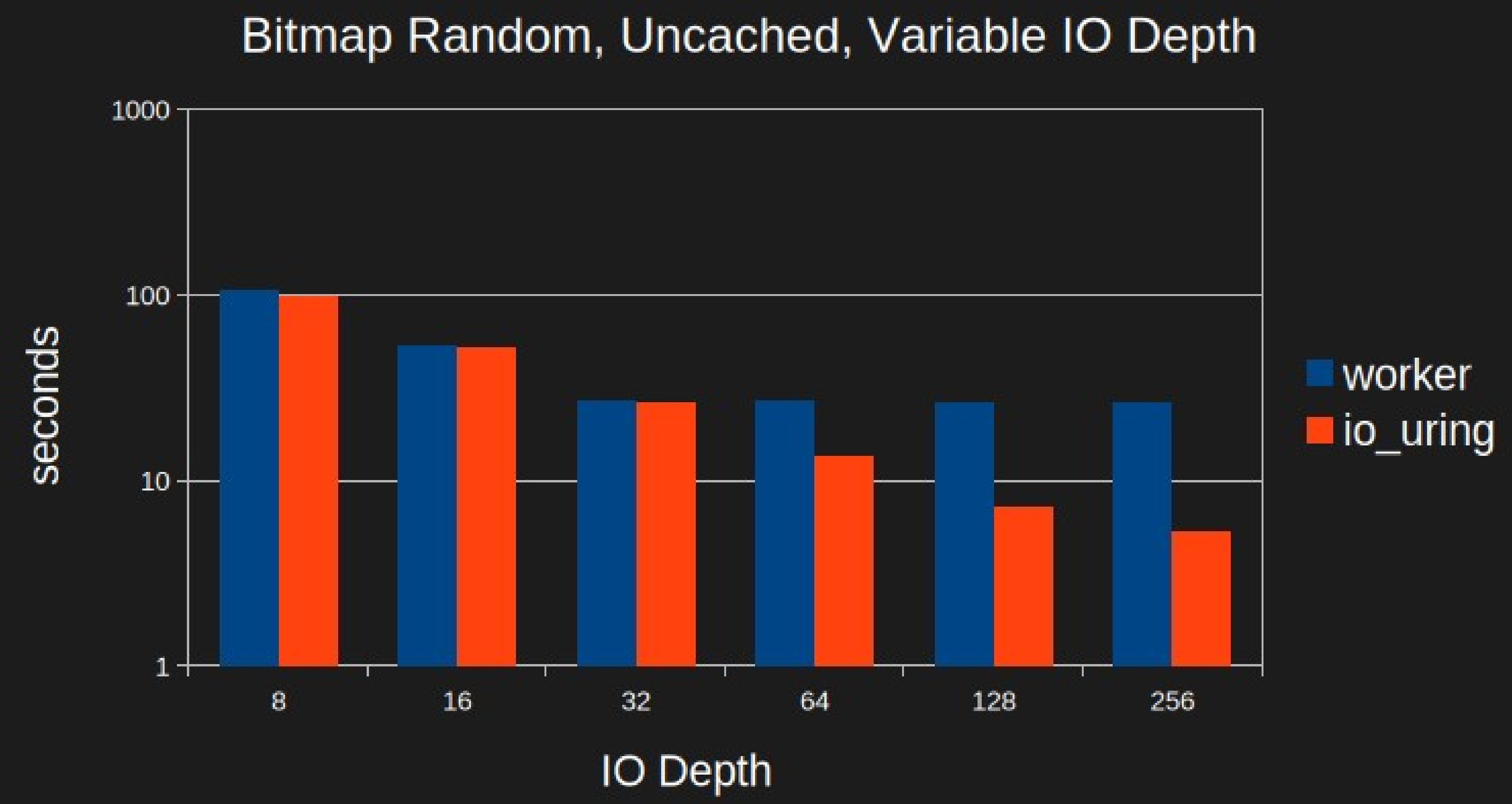
Microsoft

18: AIO for Bitmap Heap Scan



Checksums
not parallelized

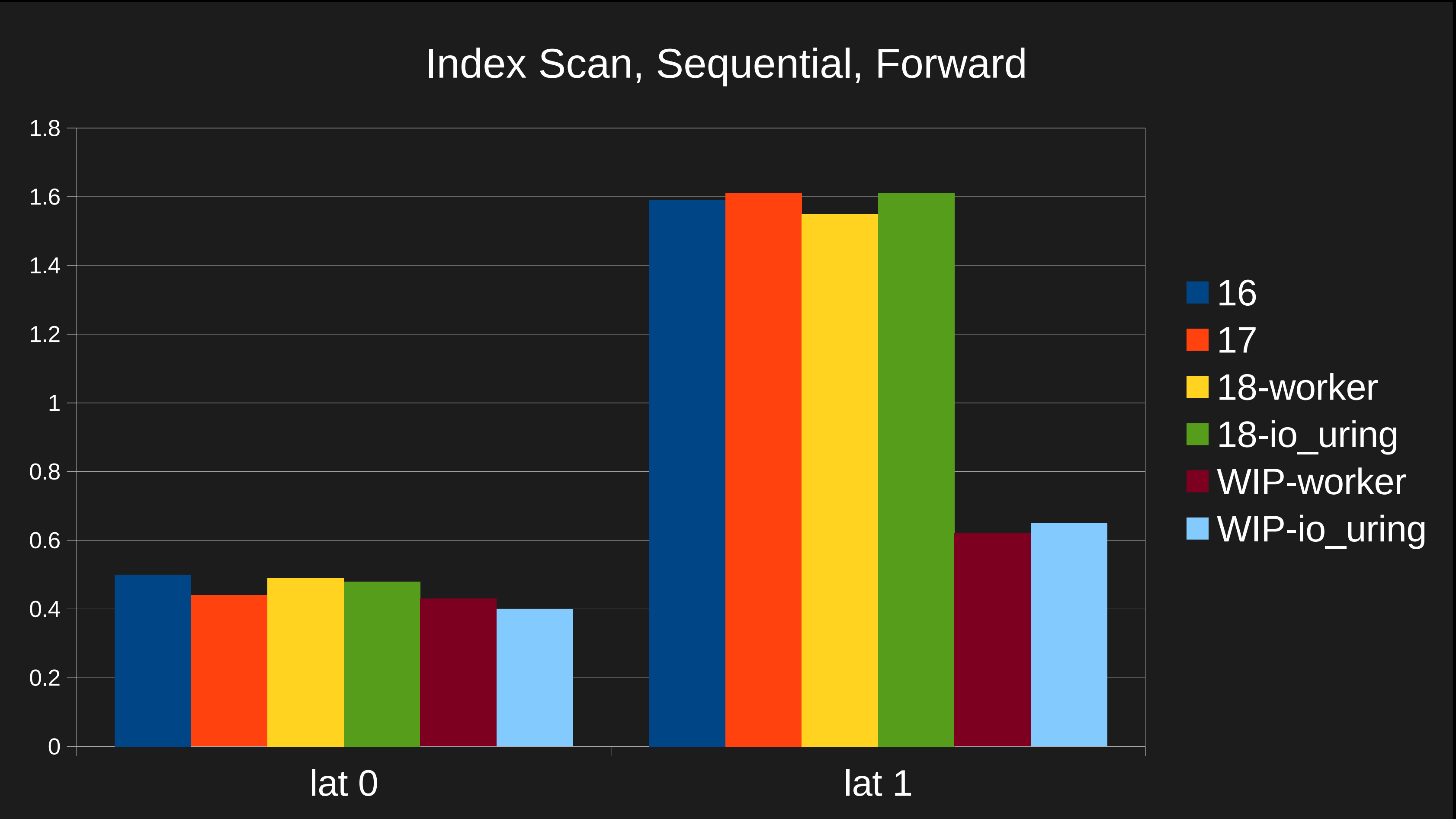
IO Depth vs io_method



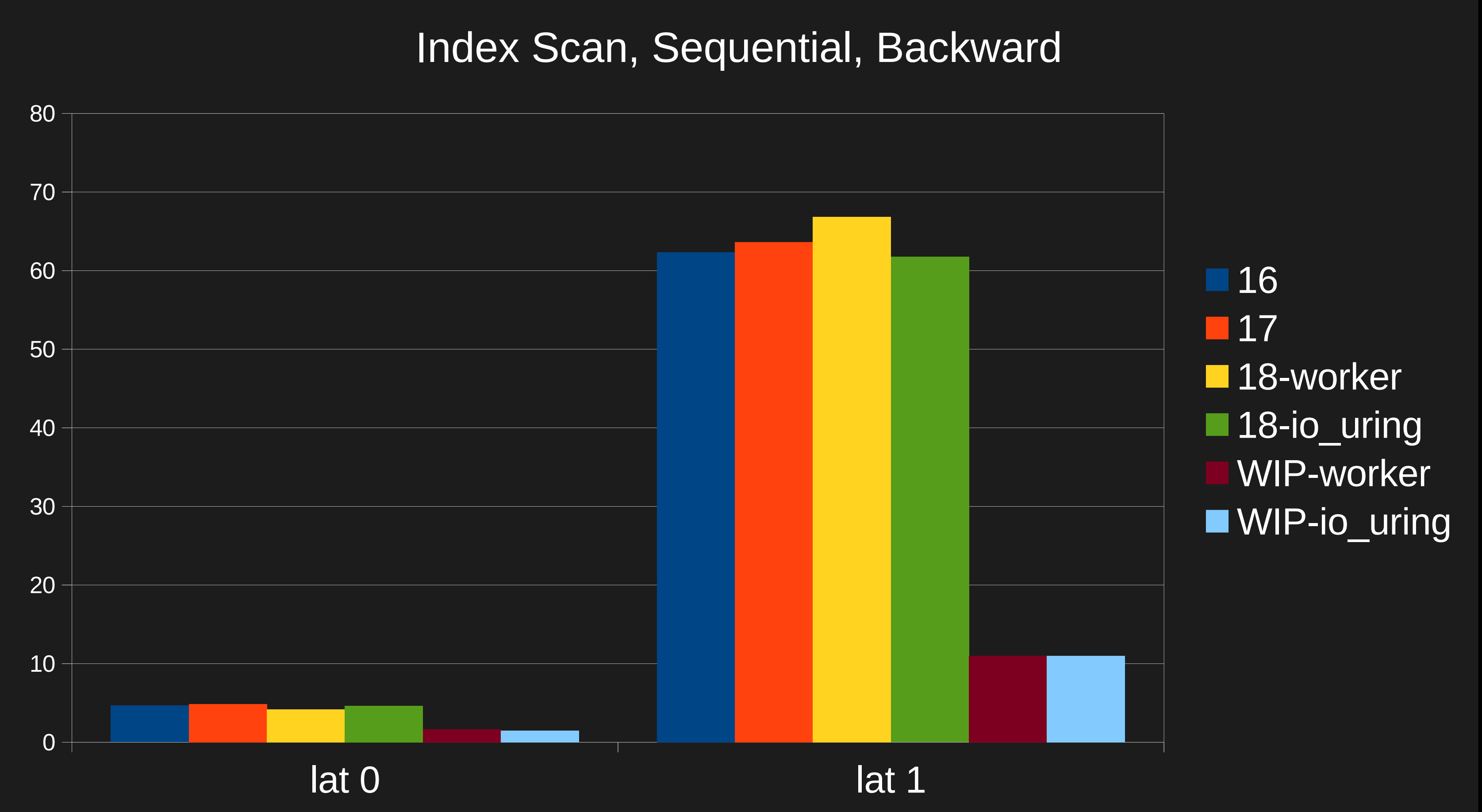
19? 20?: Index Readahead

- Tomas Vondra w/ help from Peter Geoghegan
- Much harder than already-existing AIO users
- Other performance benefits plausible
- Some regression potential too

19? 20?: Index Readahead

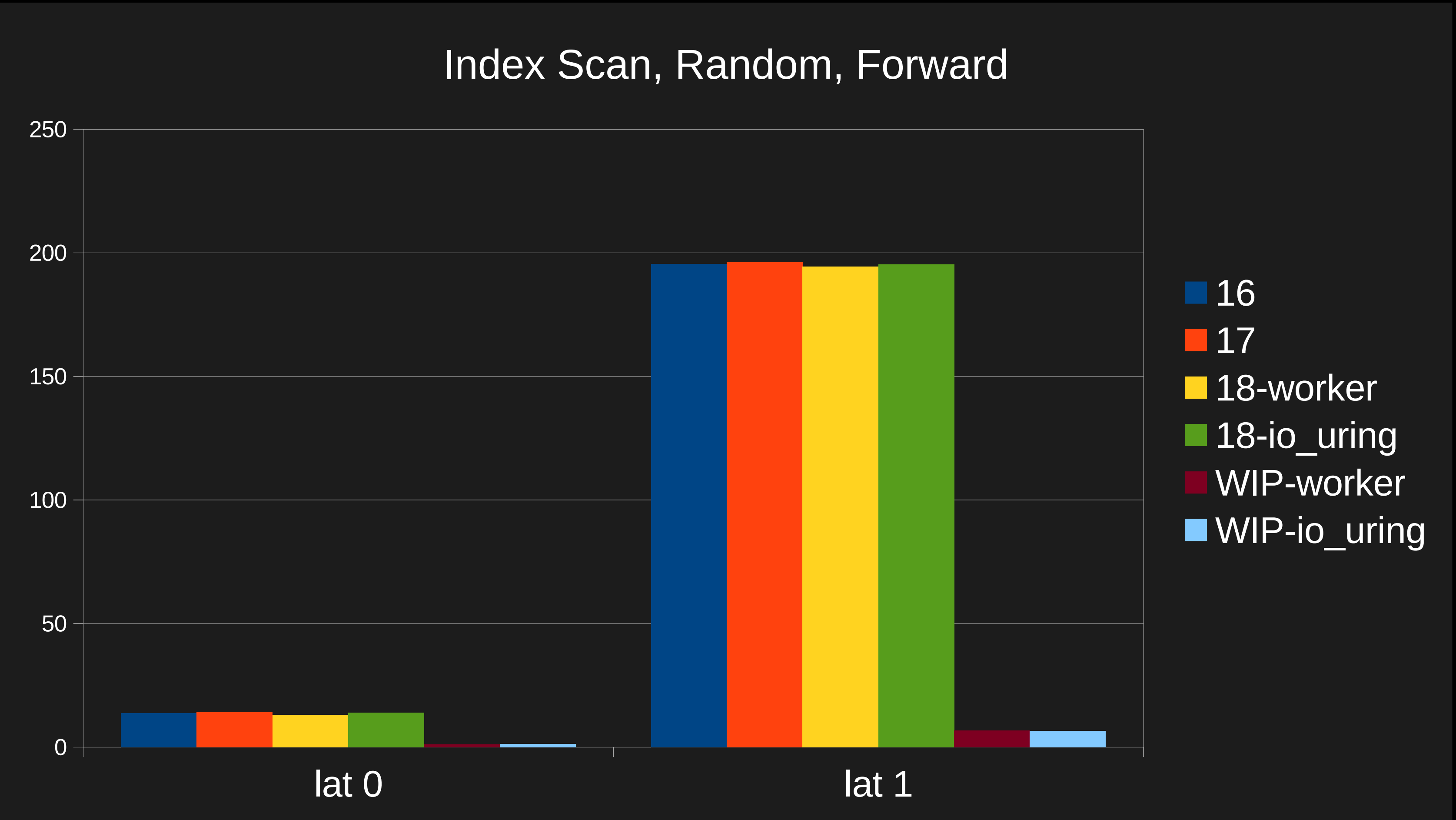


19? 20?: Index Readahead



Index Scan, Sequential, Backward

19? 20?: Index Readahead



19?: AIO writes in bgwriter & checkpointer

- Infrastructure for Buffered AIO writes required
- 2-3x checkpoint speed for sequential data
- bigger for large amounts of random data

19?: AIO for COPY & VACUUM

- Infrastructure for Buffered AIO writes required
- 2-4x speedup observable
- Bottleneck often elsewhere
  - WAL
  - index lookups

20, 21?: AIO for WAL writes

- Hard
- Huge wins possible
- Helpful for
  - Bulk load
  - Concurrent OLTP workloads
- Not helpful for
  - low concurrency OLTP

Future AIO Users

- Recovery Readahead
  - crucial for working without full-page-writes / RWF_ATOMIC
- alter database set tablespace
- create database reads (strategy file_copy) & writes
- fsyncing files at end of checkpoint
- unlinking lots of files
- ...

Other Future Work

- Other IO methods
  - Windows IOCP or io_uring
  - FreeBSD (+others?) posix_aio
- Optimize existing code
  - auto-tune number of workers
  - registered buffers for io_uring
- Integrate async network IO

AIO in Postgres 18 and beyond

Andres Freund
PostgreSQL Developer & Committer
Email: andres@anarazel.de
Email: andres.freund@microsoft.com

https://anarazel.de/talks/2025-09-30-pgconf-nyc-aio-in-PG-18-and-beyond/aio-in-PG-18-and-beyond.pdf