AIO in PostgreSQL 18

Tomas Vondra <tomas@vondra.me> / https://vondra.me

OpenAlt 2025, November 1-2, Brno

Tomas Vondra

- Postgres engineer @ Microsoft
- https://vondra.me
- vondratomas@microsoft.com
- tomas@vondra.me
- office hours
- ...

Prague events

Prague PostgreSQL Developer Day 2026

January 27-28

CfP (closes November 14)
https://cfp.p2d2.cz/2026/

looking for sponsors & partners

Prague PostgreSQL Meetup

https://www.meetup.com/prague-postgresql-meetup

Agenda

- what's AIO
- a bit of history
- what's in PG18
- what's being worked on (PG19?)
- tuning / open issues

https://www.postgresql.eu/events/pgconfeu2025/schedule/session/7001-aio-in-pg-18-and-beyond/

https://www.youtube.com/@pgeu/videos

https://www.pgevents.ca/events/pgconfdev2025/schedule/session/430-what-went-wrong-with-aio/

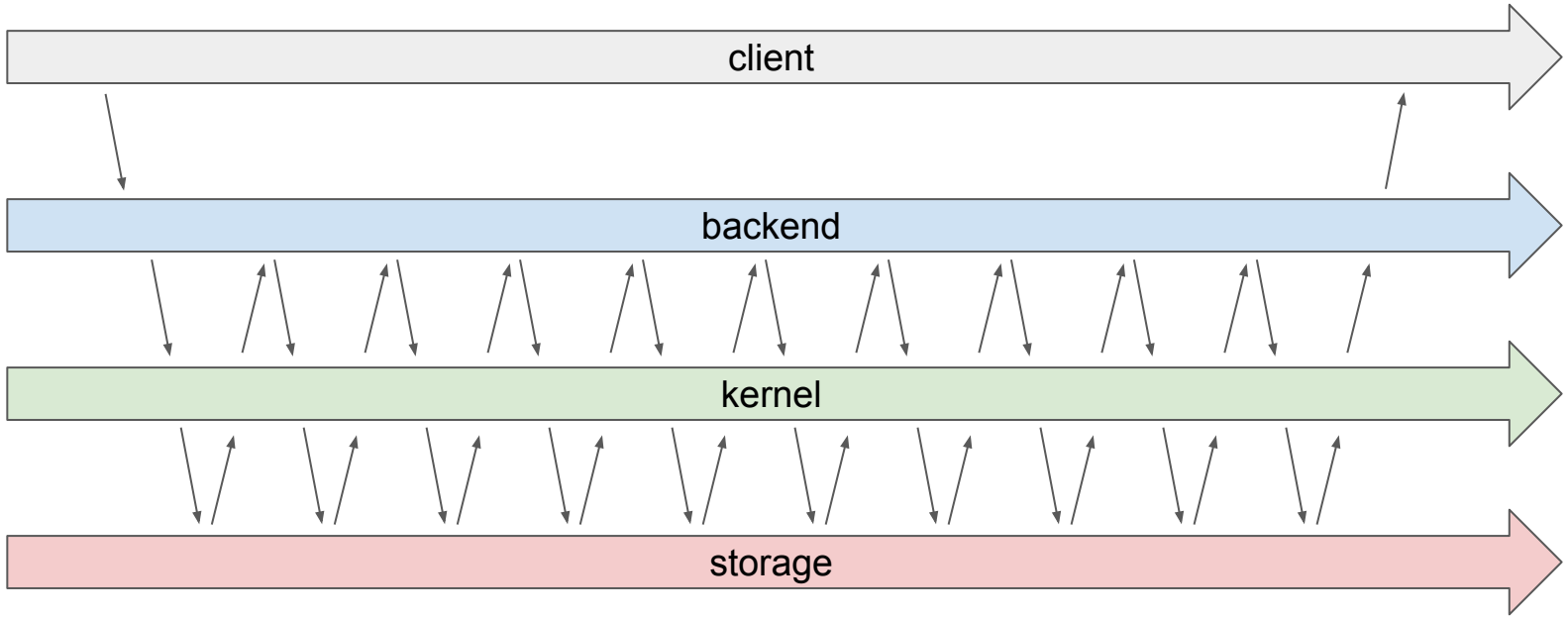
https://www.youtube.com/watch?v=GR5v9DHiS8w

What's AIO?

- AIO = asynchronous input output
- ability to issue I/O requests ahead of time
- important for
  - high-latency storage (e.g. in the cloud)
  - storage requiring high I/O depth
- requires "predicting future"
  - example: kernel read-ahead (for sequential access)

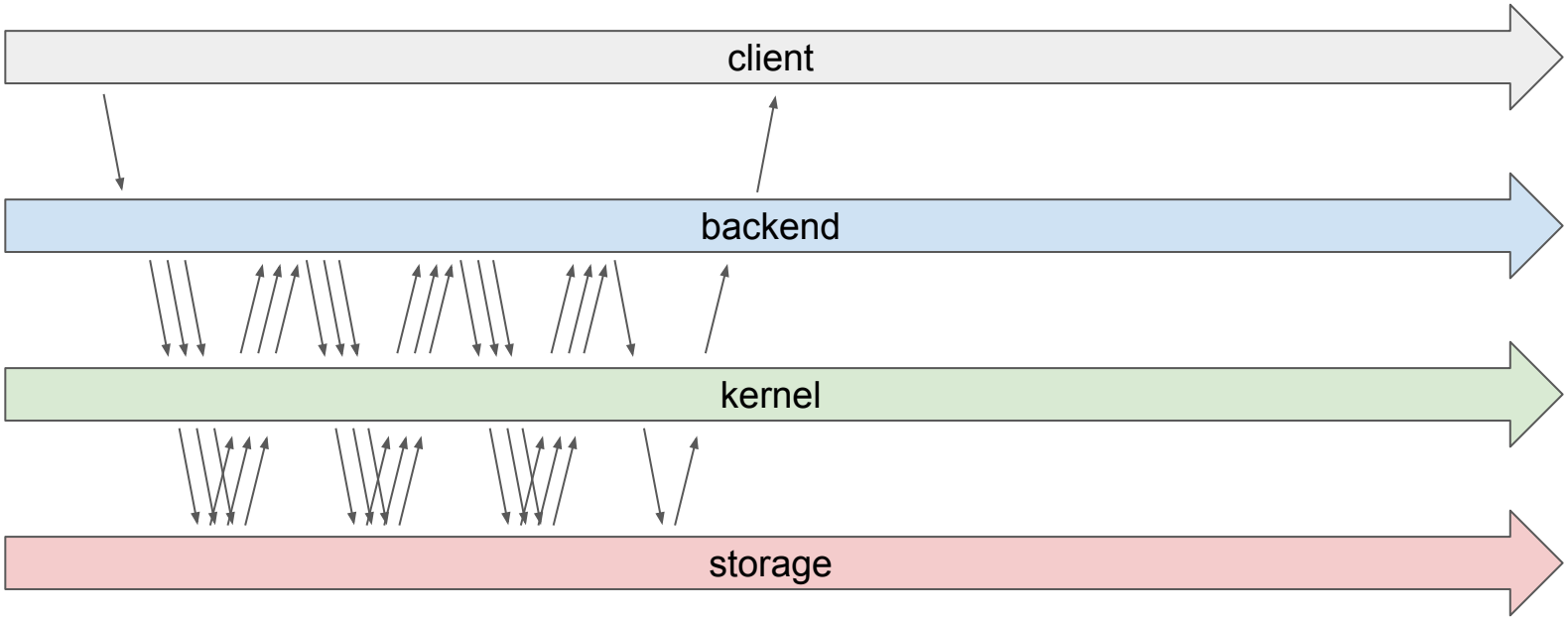
What's AIO?

- synchronous I/O (traditional)



What's AIO?

- asynchronous I/O (new in PG18)



AIO progress

older versions

- synchronous I/O
- posix_fadvise prefetching

16

- Buffer Manager 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

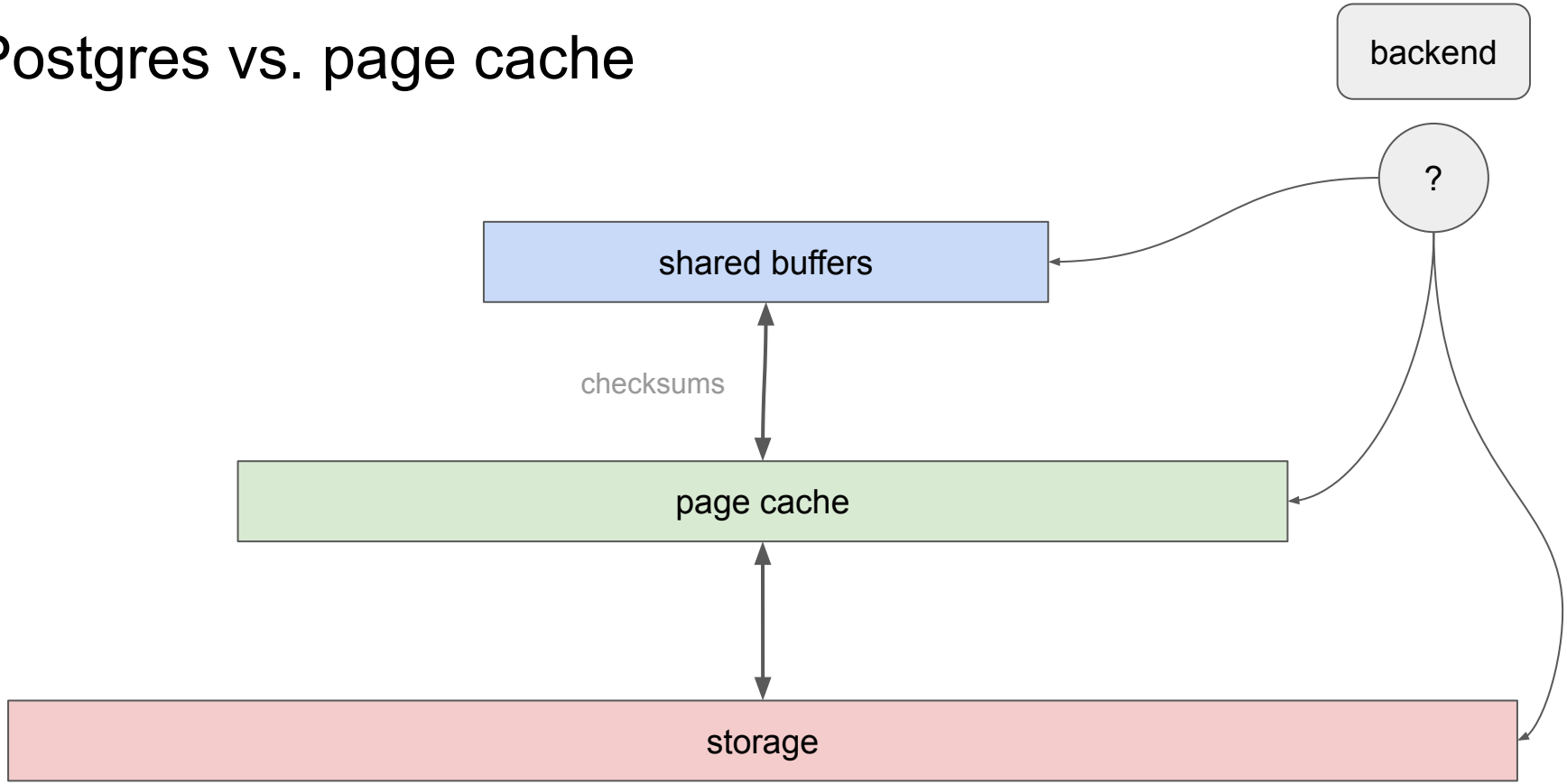
future (19, 20, ...)

- writes (relations, WAL, ...)
- streamify index scans

Why it took so long?

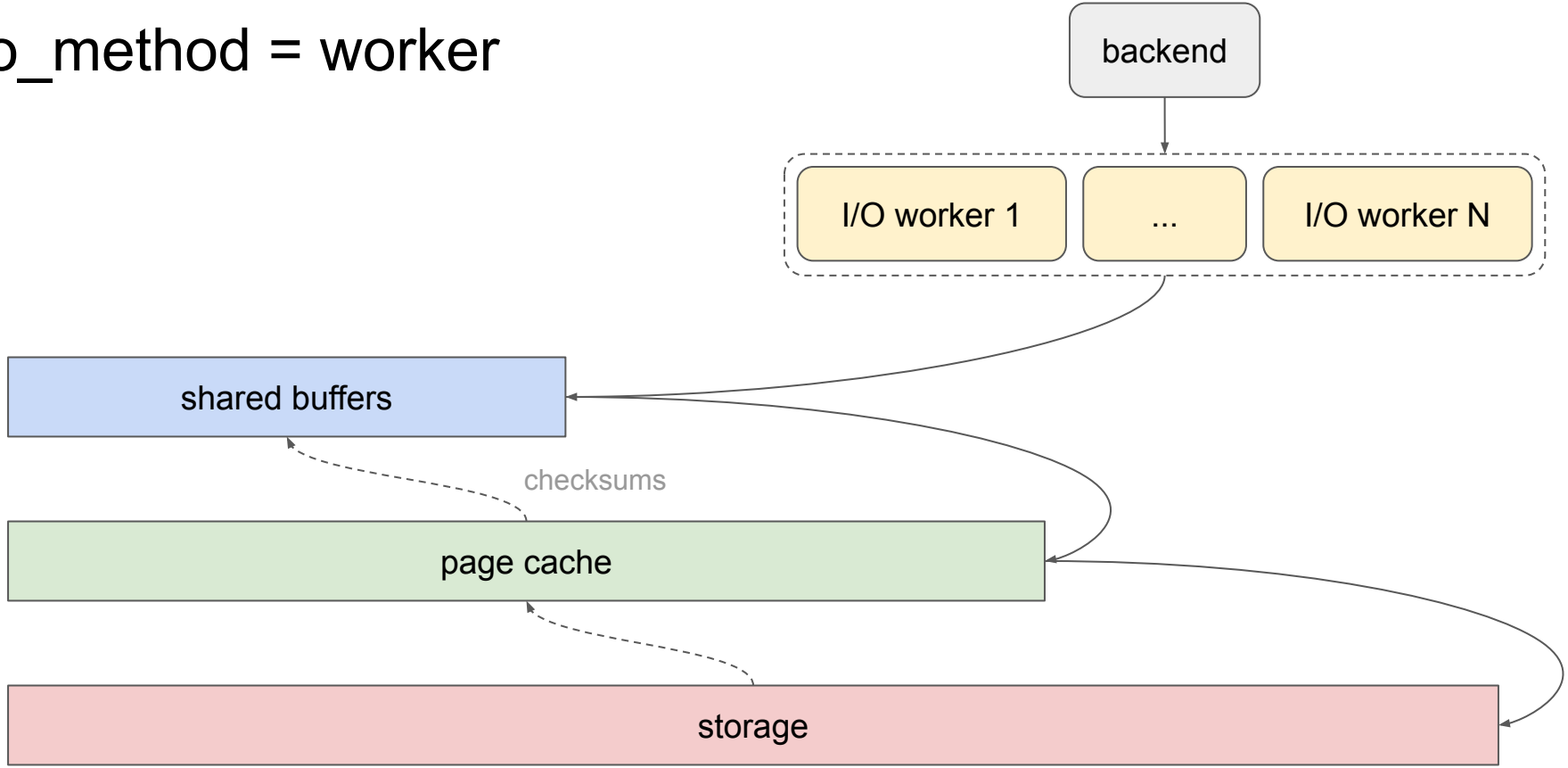
- other DBs did that earlier
  - sometimes with direct I/O or raw devices
- so why not Postgres?
  - it's a lot of work
  - requires a lot of dev time to get right
  - other stuff had better cost/benefit
- aio libraries exist ...
  - but some are relatively new
  - problematic compatibility / platform support
  - some thread based, ...

Postgres vs. page cache

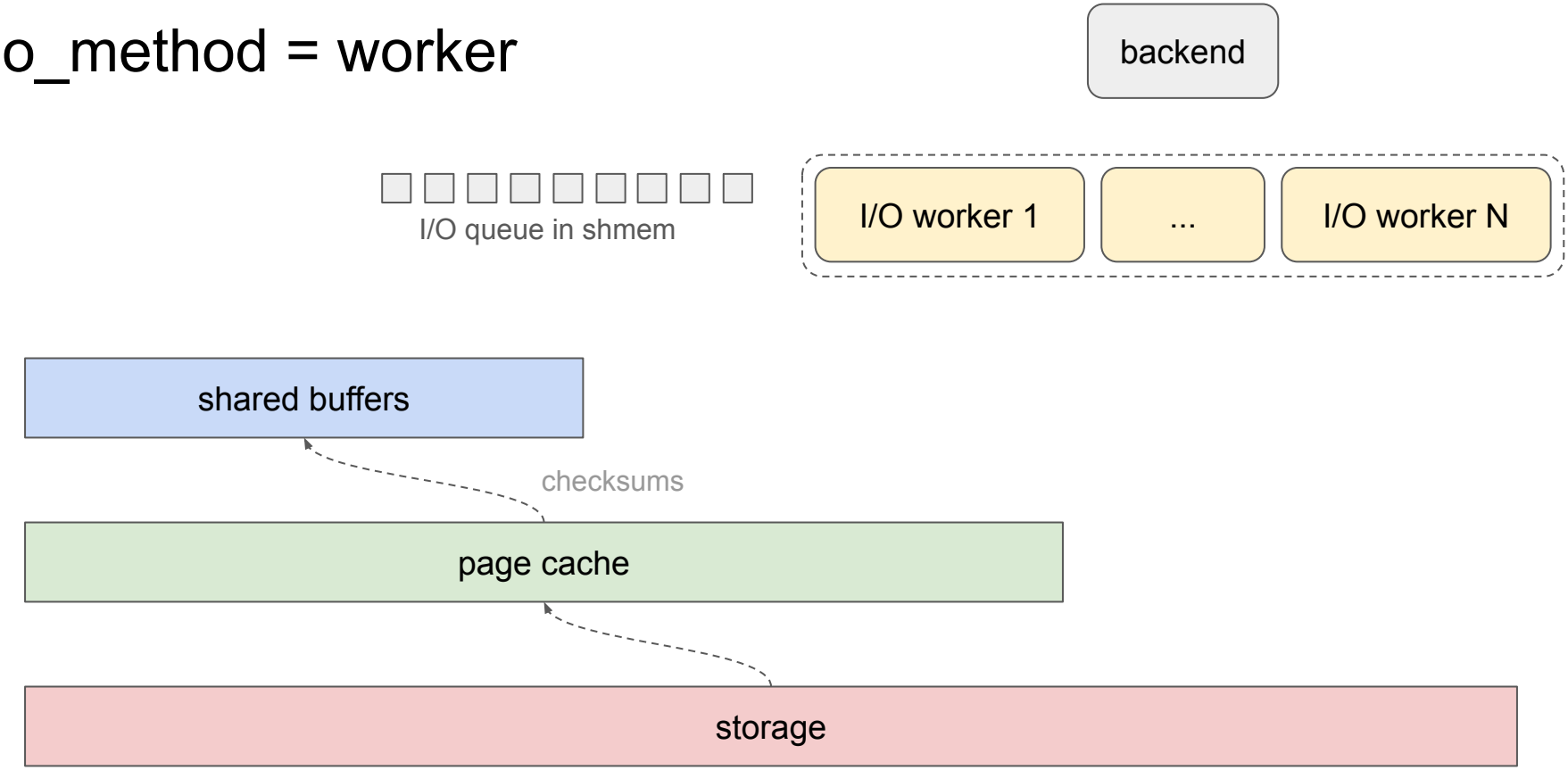


io_method = ?

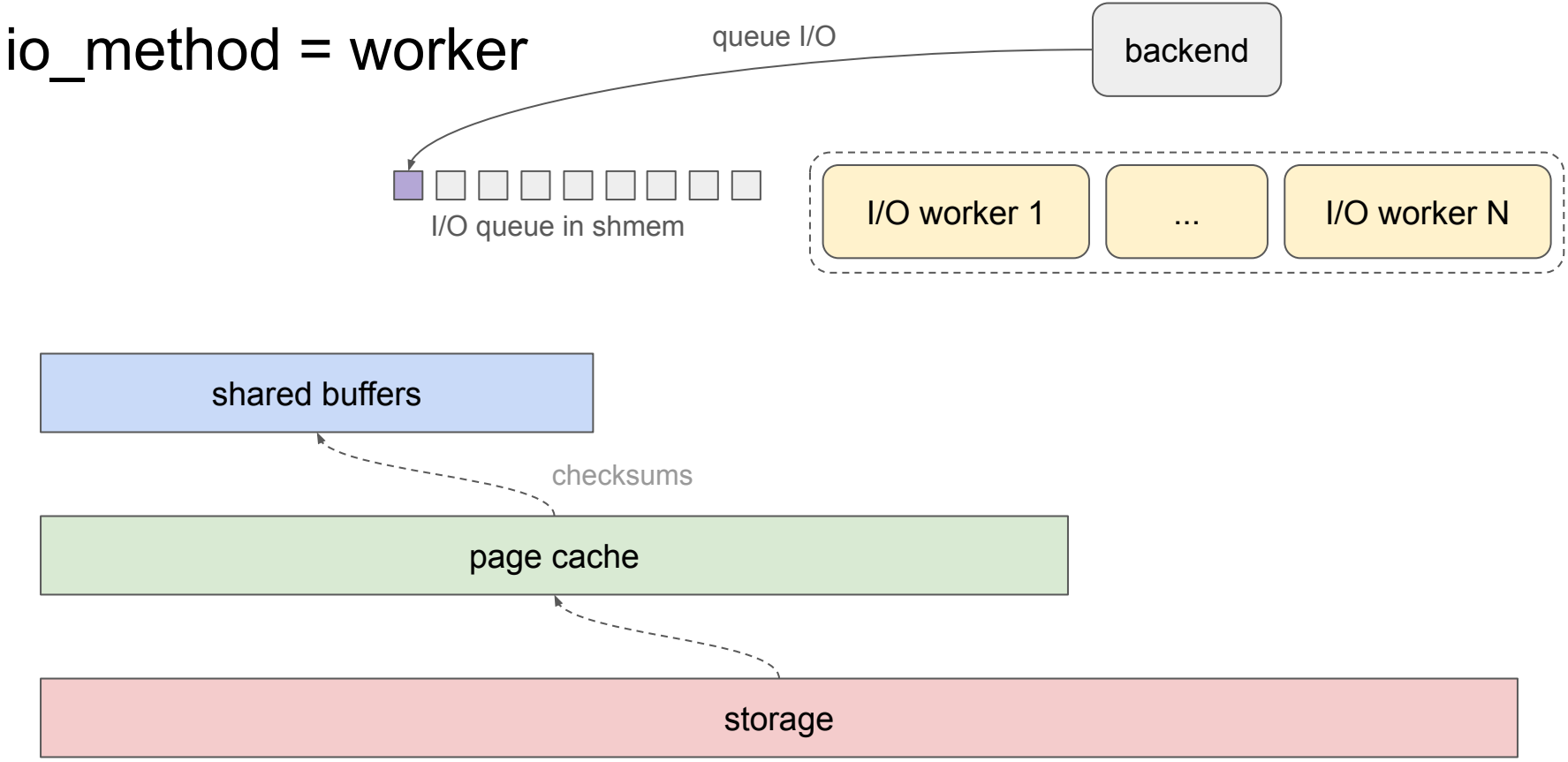
io_method = worker



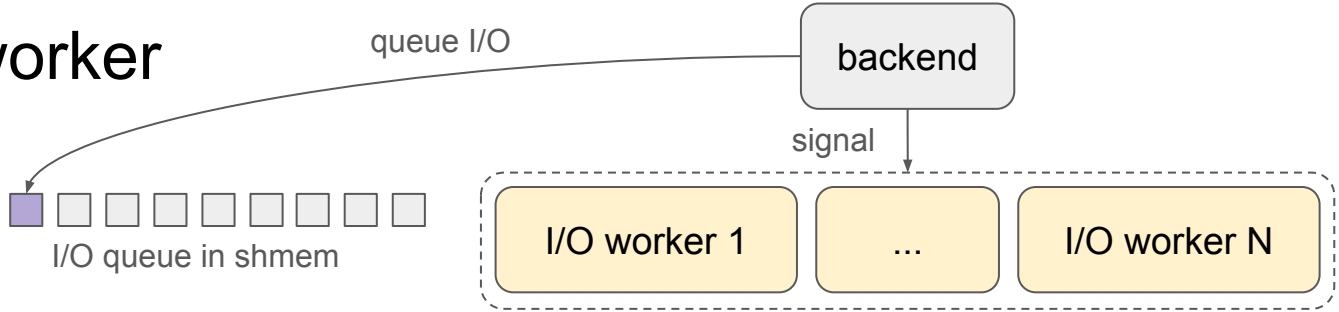
io_method = worker

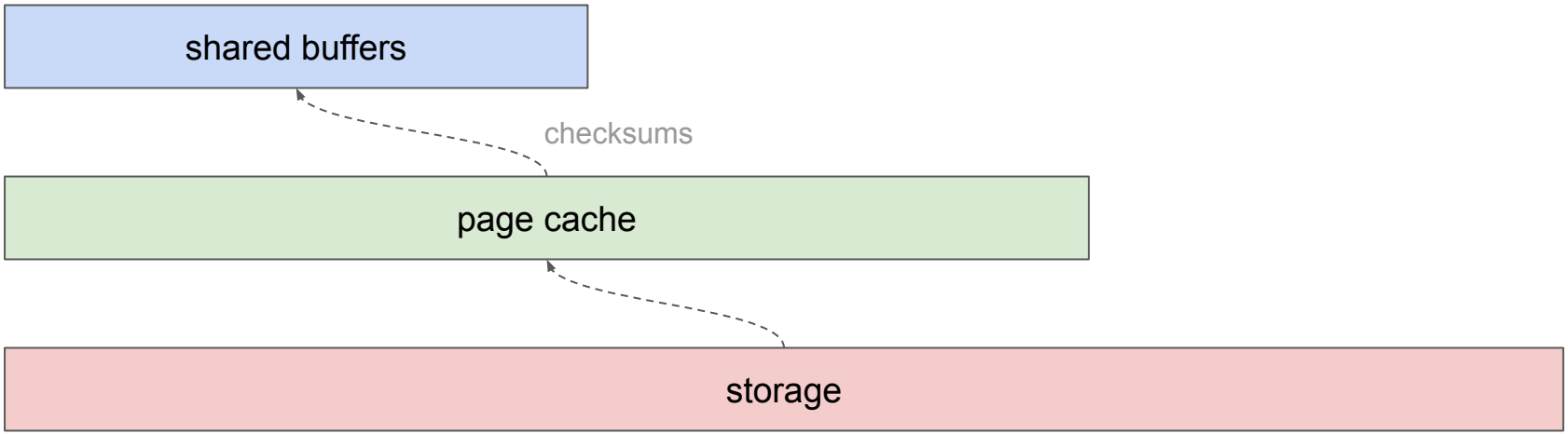


io_method = worker

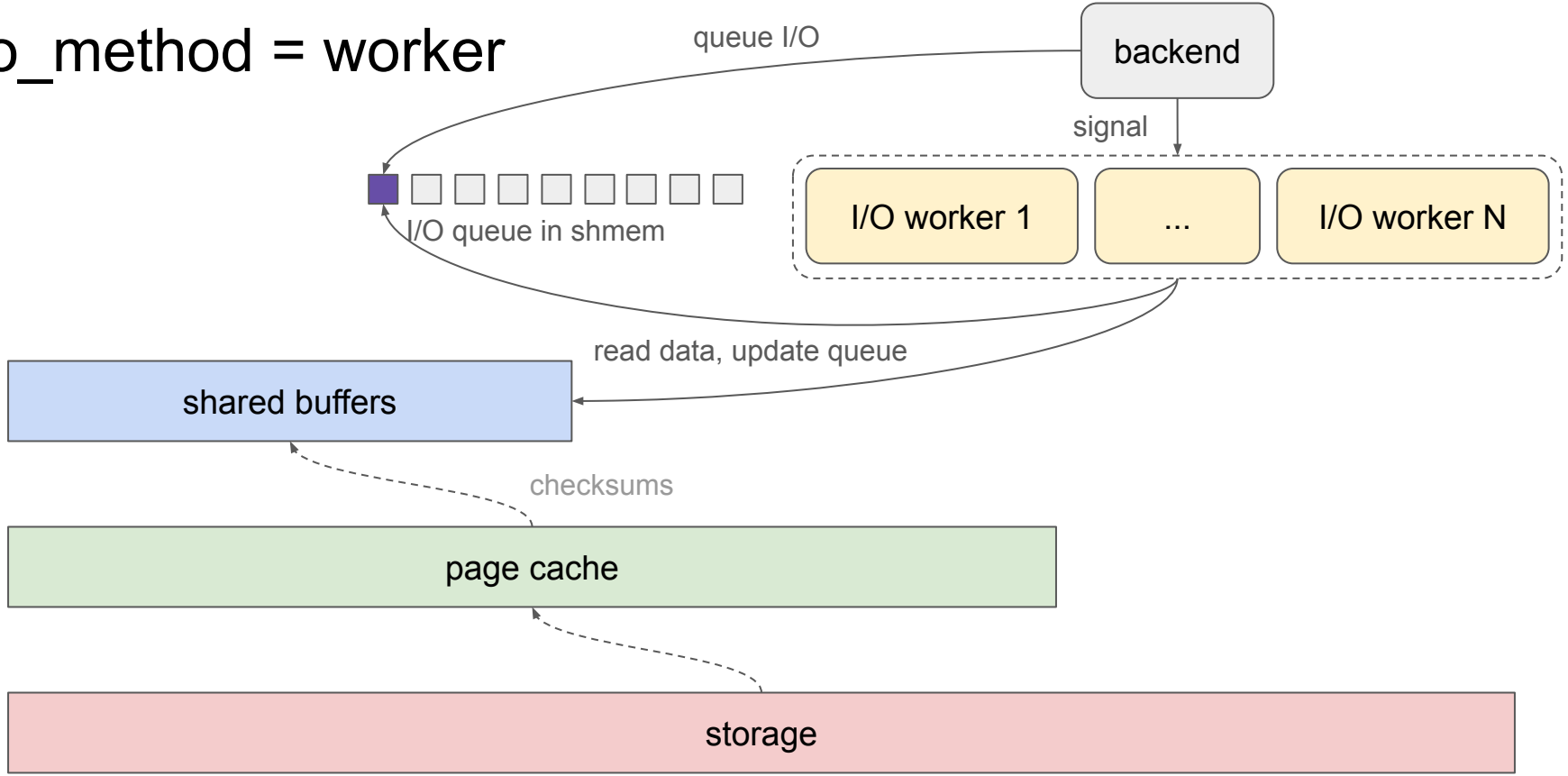


io_method = worker

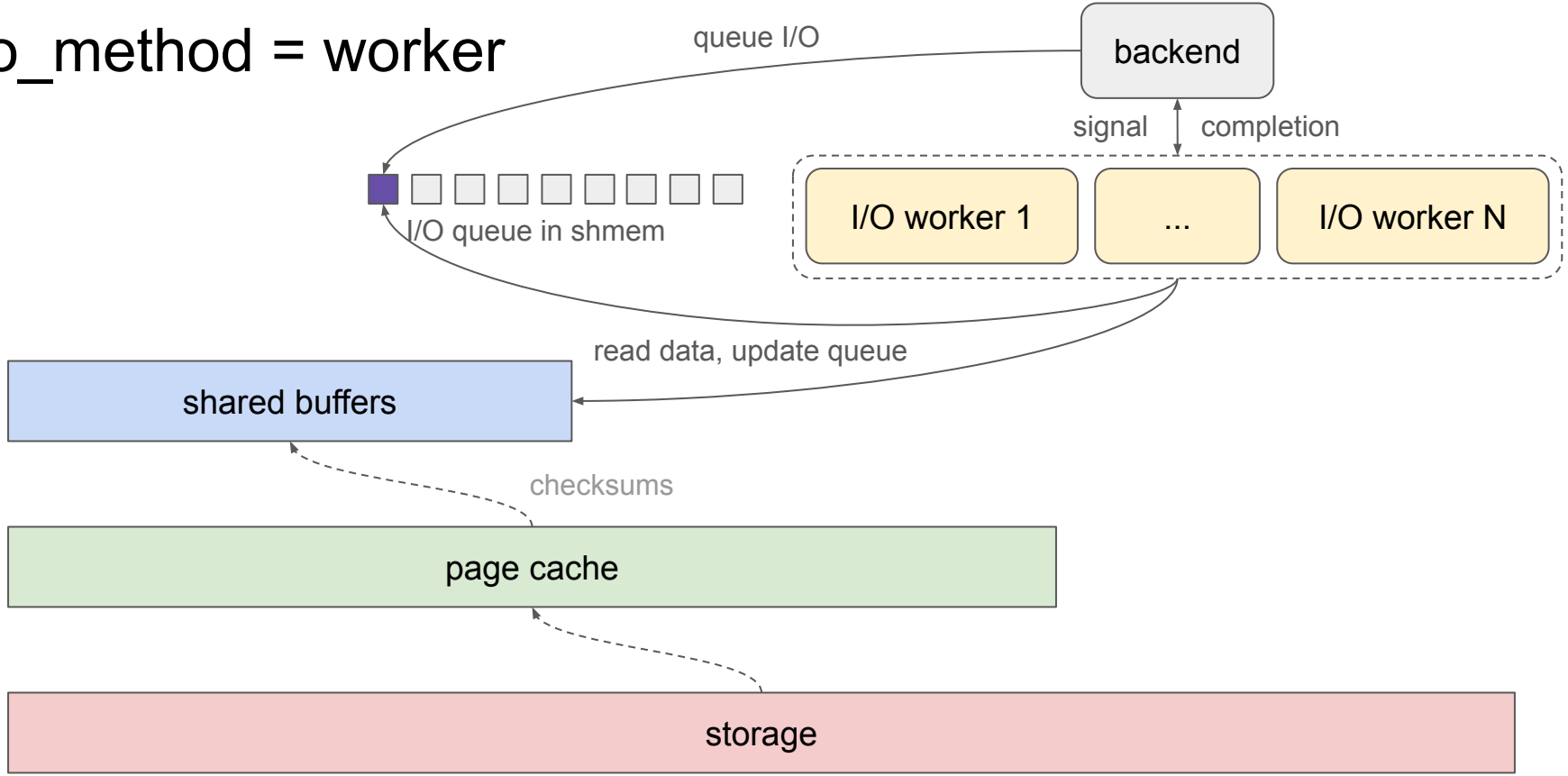




io_method = worker



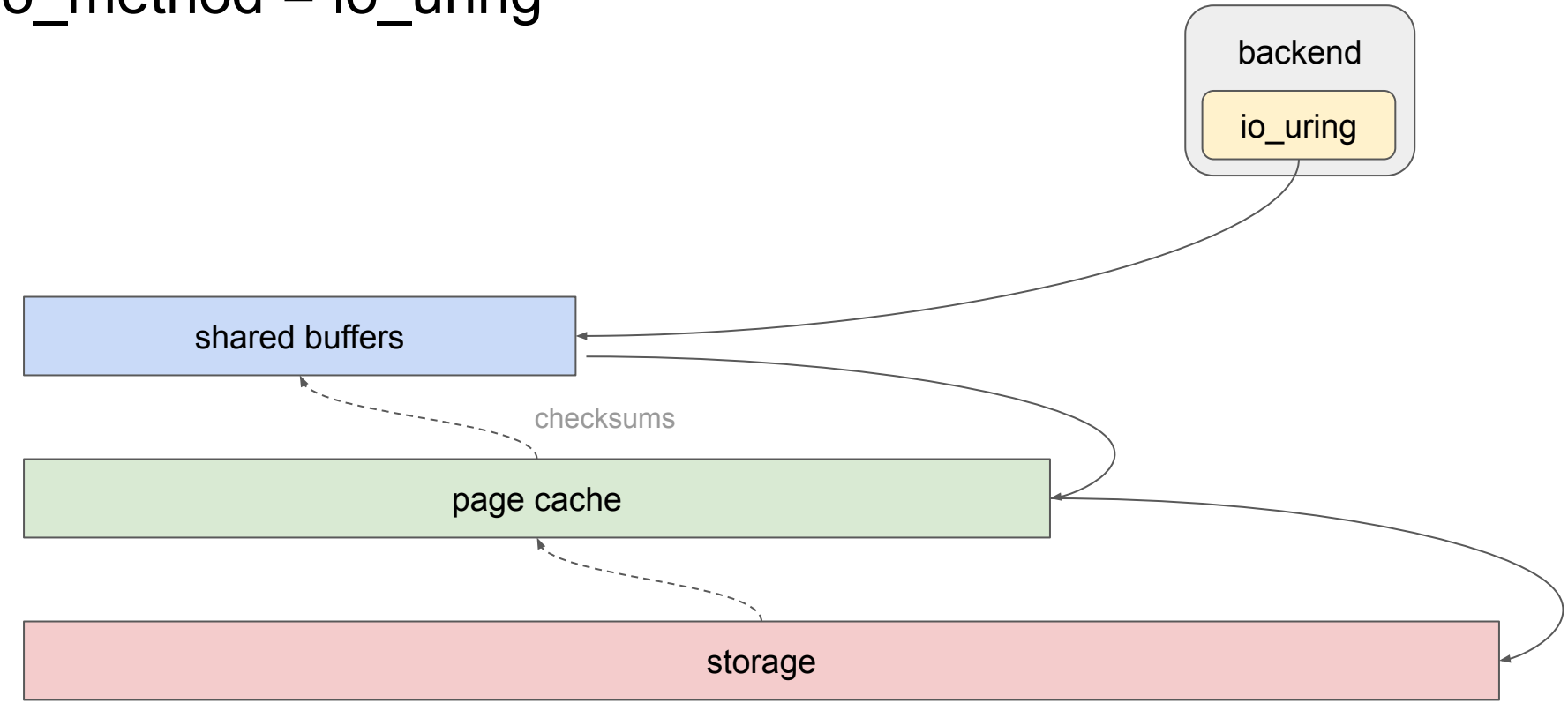
io_method = worker



io_method = worker

- portable
  - fully under our control
- parallelizes checksums, memory copy
- limited I/O depth
  - particularly with high latency storage
- IPC based on signals (may be bottleneck)
  - mostly just rare pathological cases (page-at-a-time)
- number of workers controlled by io_workers
  - default 3, maximum 32
  - global limit

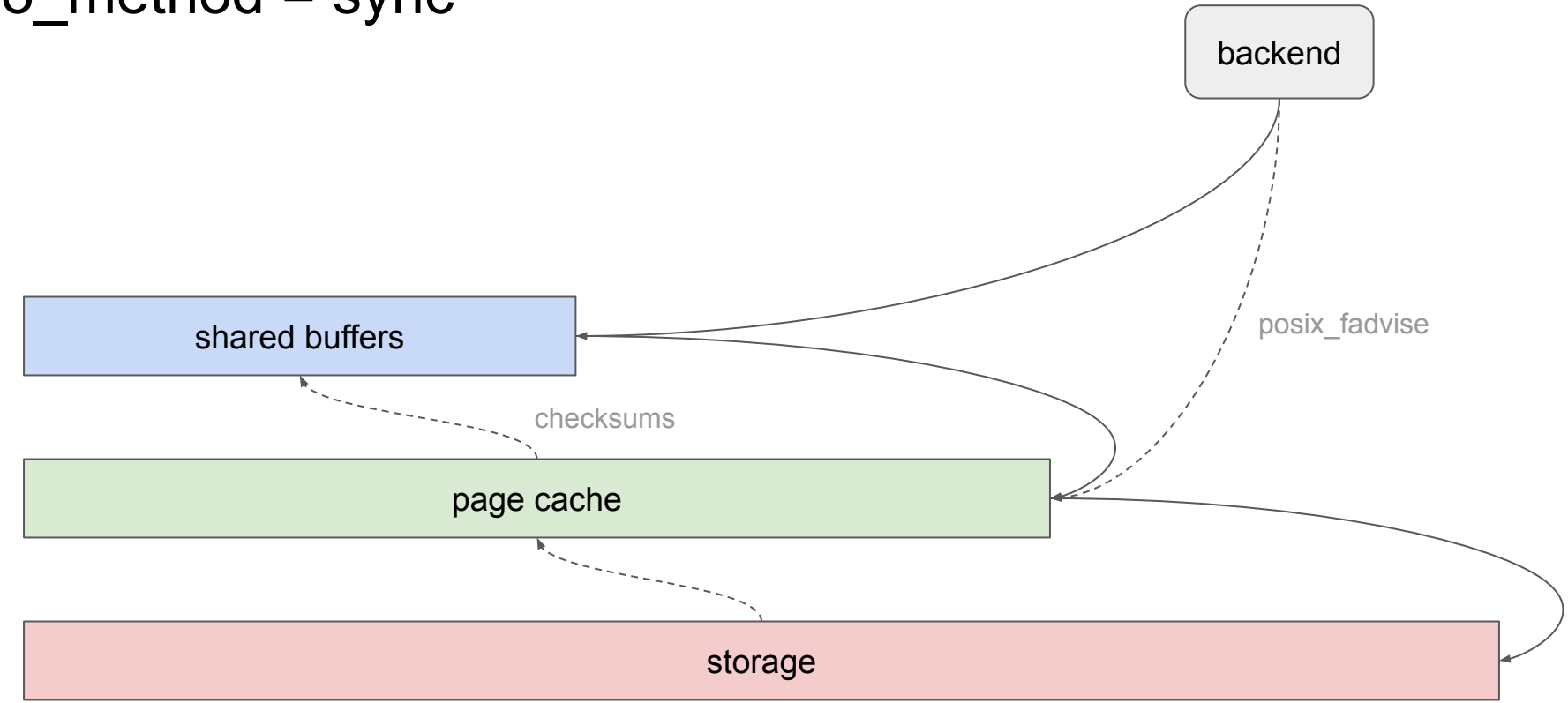
io_method = io_uring



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 additional tuning
  - file descriptor limits

io_method = sync



io_method = sync

- “safety net”
- behaves as close as realistic to < 18
- doen’t "use" AIO
  - goes through stream
  - but synchronously
- prefetch using posix_fadvise
  - only to page cache, not shared buffers
- incompatible with direct I/O

Which io_method is best?

- none
- io_method=worker
  - default
  - good: compatibility, can parallelize checksums
  - bad: io_workers=3 a bit too low (max 32 a bit low too - iodepth=32)
- io_method=io_uring
  - Linux only, very modern / efficient
  - all in a single process, no parallel checksums
  - can do much deeper IO queues

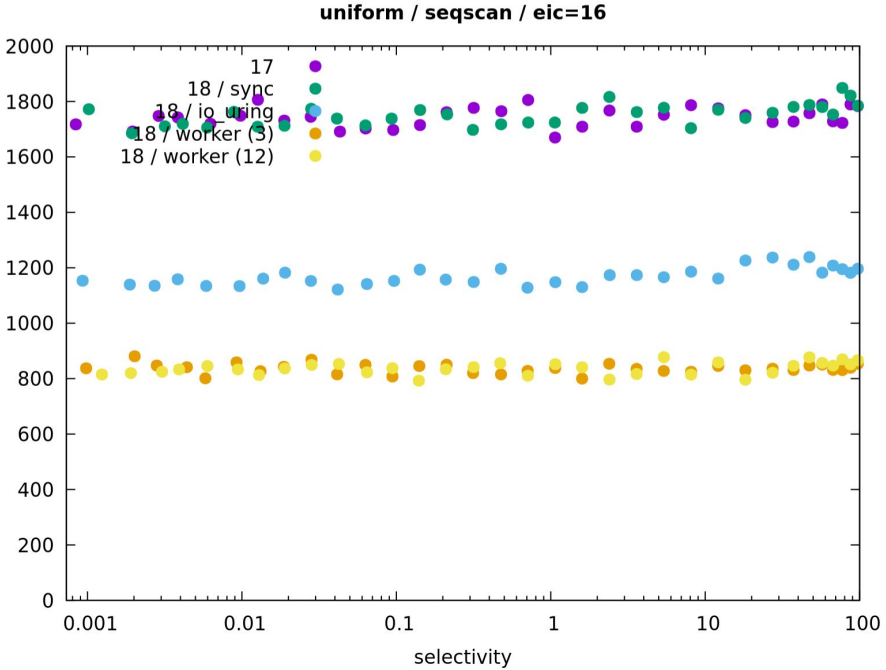
https://vondra.me/posts/tuning-aio-in-postgresql-18/

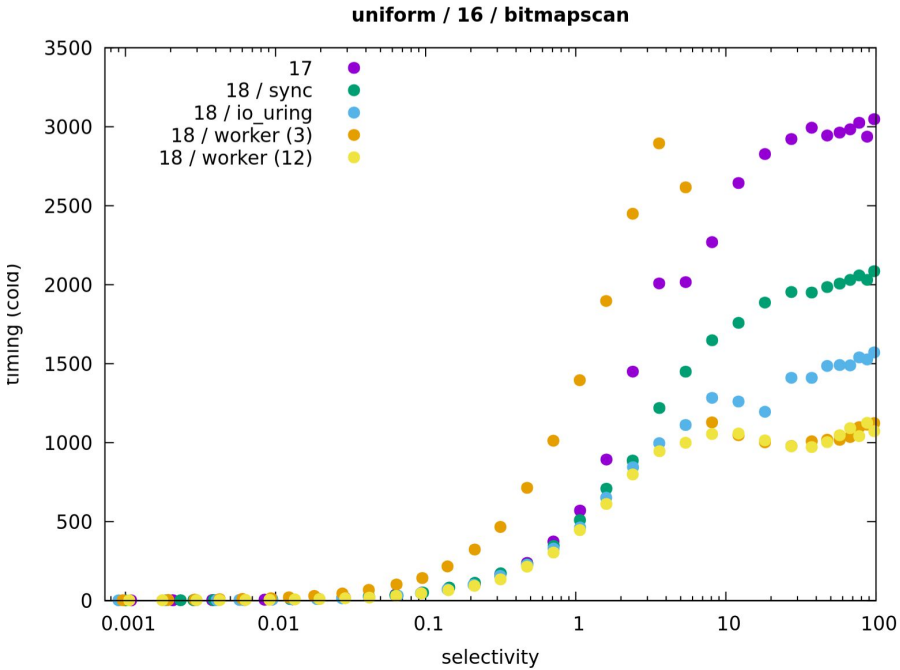
performance

PG18: When can AIO help?

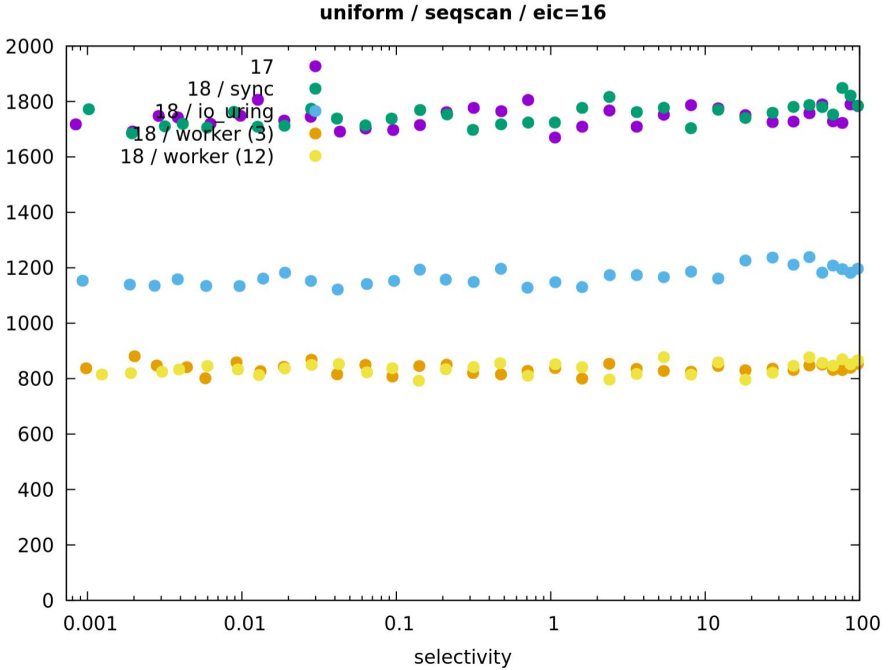
- IO bound
  - track_io_timing
  - EXPLAIN (ANALYZE, BUFFERS)
- only for reads
- foreground: seqscan, bitmap heap scan
- background: vacuum
- Just the absolute basics!

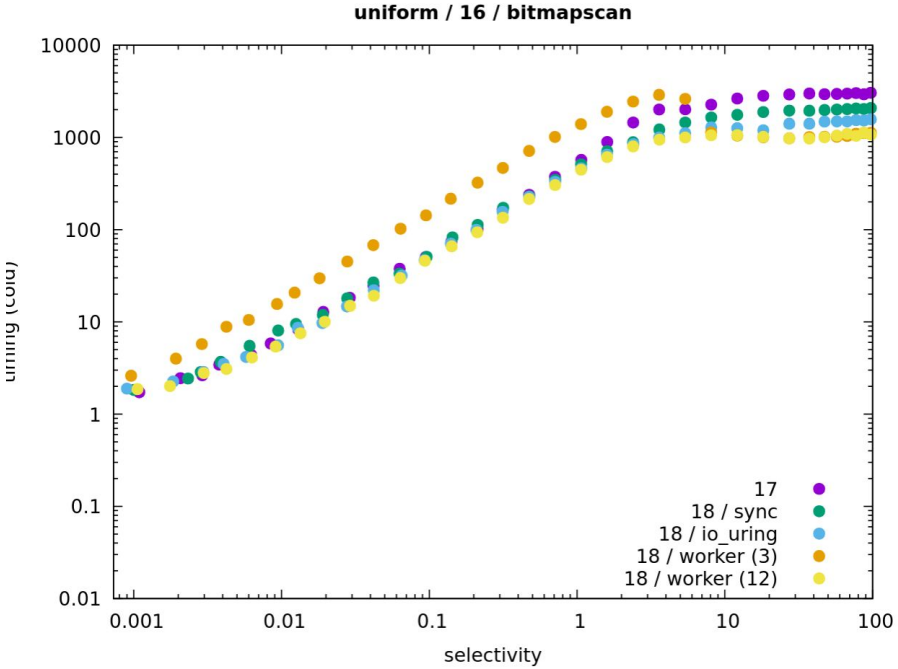
https://github.com/tvondra/iomethod-tests/





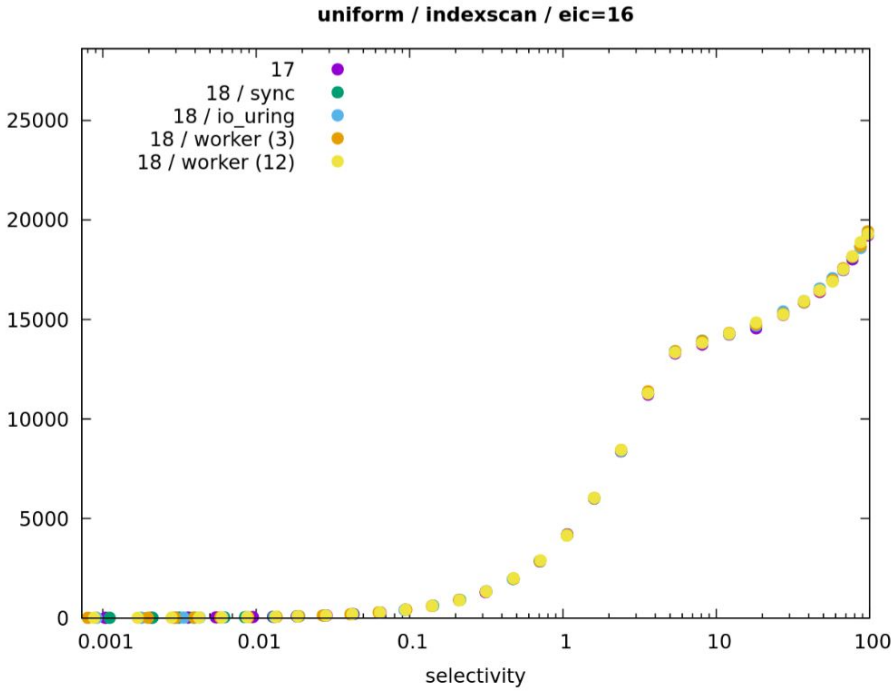
https://github.com/tvondra/iomethod-tests/

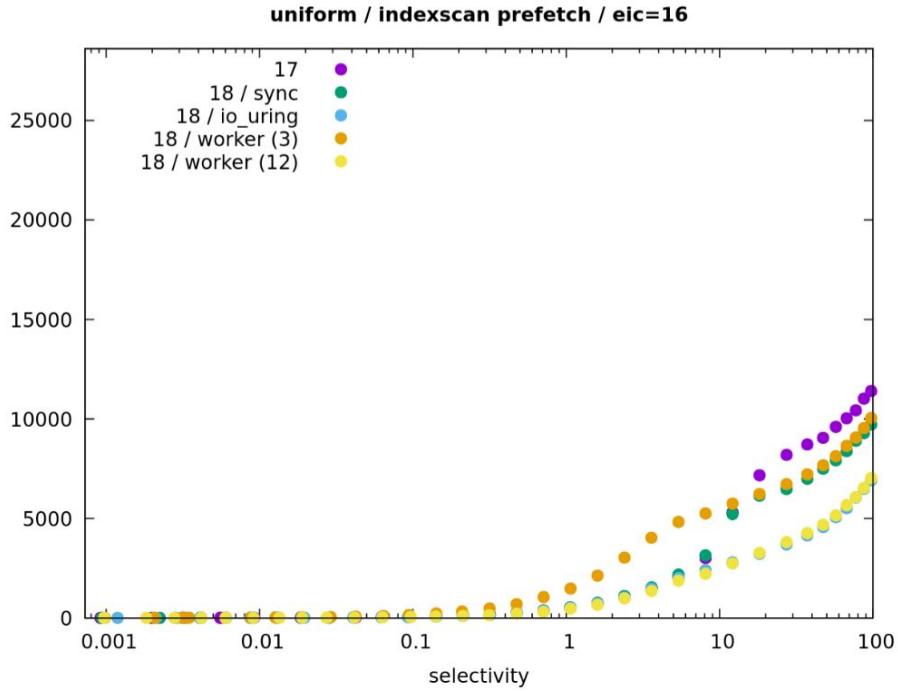




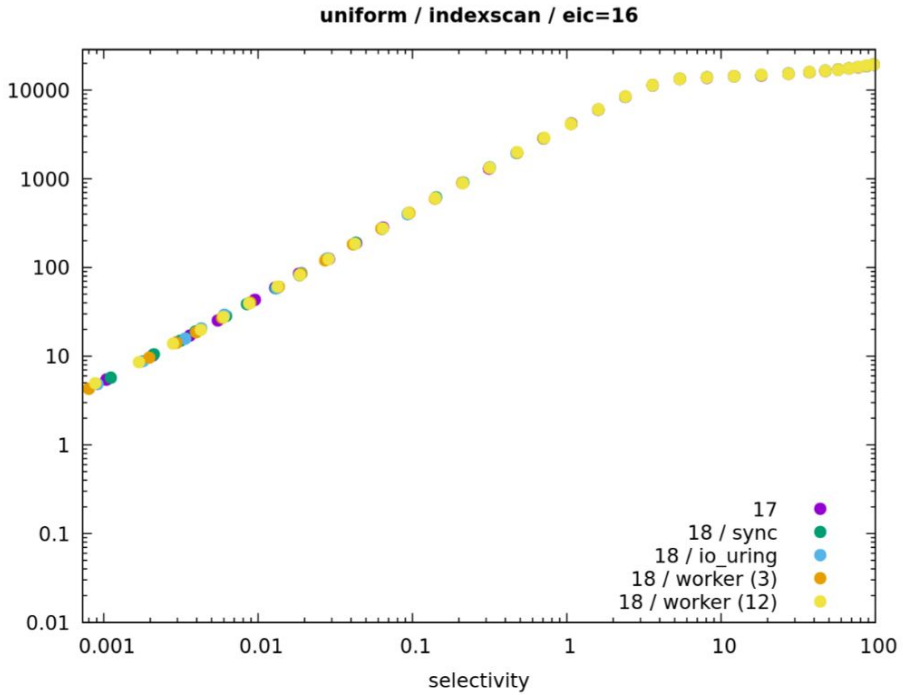
Future

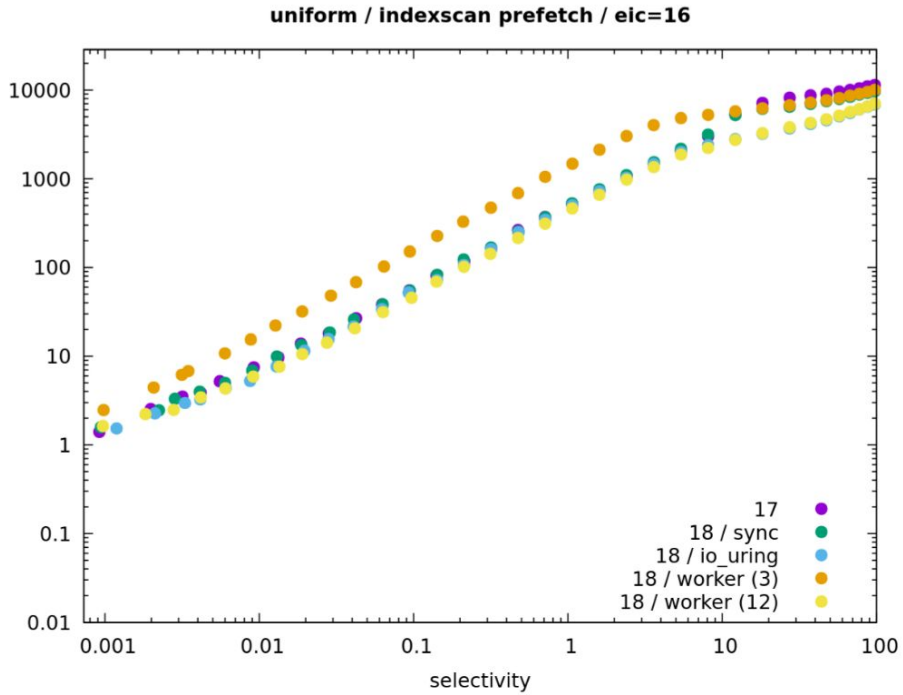
AIO / index prefetching (PG19?)





AIO / index prefetching (PG19?)





AIO writes (PG19? PG20?)

- 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
- AIO for COPY & VACUUM
  - Infrastructure for Buffered AIO writes required
  - 2-4x speedup observable
- Bottleneck often elsewhere
  - WAL (COPY & VACUUM)
  - index reads during insertion (COPY)

AIO writes (PG20? PG21?)

- AIO for WAL writes
  - hard
  - huge wins possible
- Helpful for
  - bulk load, VACUUM
  - concurrent OLTP workloads
- ● Not helpful for
  - low concurrency OLTP

more future stuff

- 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

a lot more can be done ...

- https://anarazel.de/talks/2025-10-23-pgconf-eu-aio-in-PG-18-and-beyond/aio-in-PG-18-and-beyond.pdf
- https://wiki.postgresql.org/wiki/AIO

Q&A