Free-Threaded Python 3.13

What changed in the free-threaded build, and what do benchmarks show so far?

disco LATIN · VERB /ˈdɪs.koː/

“I learn; I become acquainted with.”

from discere — to learn

Free-threading replaces the GIL with per-object locks.

- Biased, immortal, and deferred refcounting plus mimalloc let threads run in parallel.

The single-threaded cost fell sharply across releases.

- ~40% slower on 3.13, ~10% on 3.14, ~1.2% in one 3.14t test.

Single-threaded penalty vs multi-threaded scaling.

Metric	Single-threaded	Multi-threaded
3.13 free-threaded vs GIL	~40% slower	up to 4x faster

Parallel workloads on independent data scale well.

- Up to ~4x execution-time reduction in a 3.14.2 study; ~3x in an 8-thread FOSDEM test.

Shared mutable containers become contention points.

- Per-object locks serialize access; separate copies restore scaling.

3.13 is experimental and opt-in.

- Built via --disable-gil; runs as python3.13t with the 't' ABI tag.

Ecosystem support is partial and growing.

- NumPy, pandas, SciPy, PyTorch ship free-threaded wheels; OpenCV and polars still pending.

PEP 779 sets Phase II criteria.

- A 15% performance and 20% memory target, geometric mean on pyperformance.

Sources

PEP 703 · PEP 779 · CPython 3.13/3.14 docs · CodSpeed benchmark · 2026 arXiv study (2603.04782)