dbtune

Autotuning PostgreSQL: A deep dive into server parameter tuning with agentic AI

AI-DBA: Self-Driving Databases

January 23, 2026

Dr. Luigi Nardi

Founder & CEO, DBtune

About me

Mixed background in industry and academia

Among other things:
Ph.D. CS at Sorbonne, Research Staff at Stanford, and Associate Professor in AI at Lund

Since 2020, Founder & CEO at DBtune

SAPIENZA
UNIVERSITÀ DI ROMA

cnrs

MUREX

UPMC
SORBONNE UNIVERSITÉS

LUND
UNIVERSITY

Imperial College
London

dbtune

dbtune

What

DBtune is an AI-powered database tuning service

Where

Spun out of research at Stanford University

How

Tunes for a specific workload, use case and machine

What is database performance tuning?

What is database tuning?

Keeping the database fit and responsive

- ✓ Databases change, grow and slow down
- ✓ Not all workloads and machines are the same
- ✓ **Tuning adapts a database to its current use-case, load and machine**
- ✓ It is a ‘dark-art’ yet an integral part of any DBA and developer’s job
- ✓ Tuning includes query, **server parameters***, index, OS parameter, etc.

*This talk focuses solely on agentic AI for PostgreSQL parameter tuning

Why does it matter?

Technical perspective

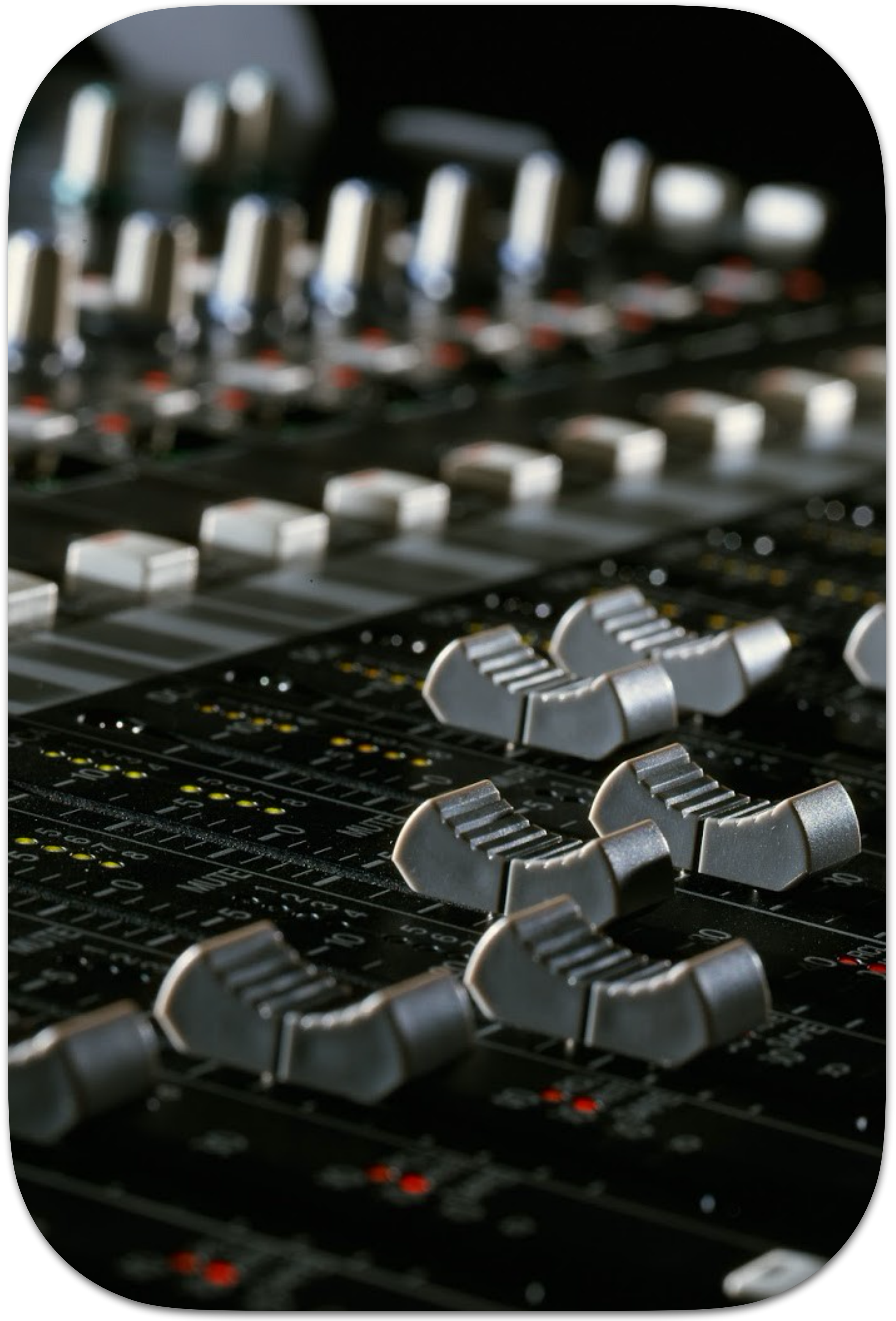
- Impacts system performance
  - Throughput and latency
- Improves scalability / stability / SLA

Business perspective

- Higher end-user satisfaction
- Optimizes infrastructure spend
- Reduces downtime
- Increases productivity
- Saves energy (ESG)

Database system parameter tuning

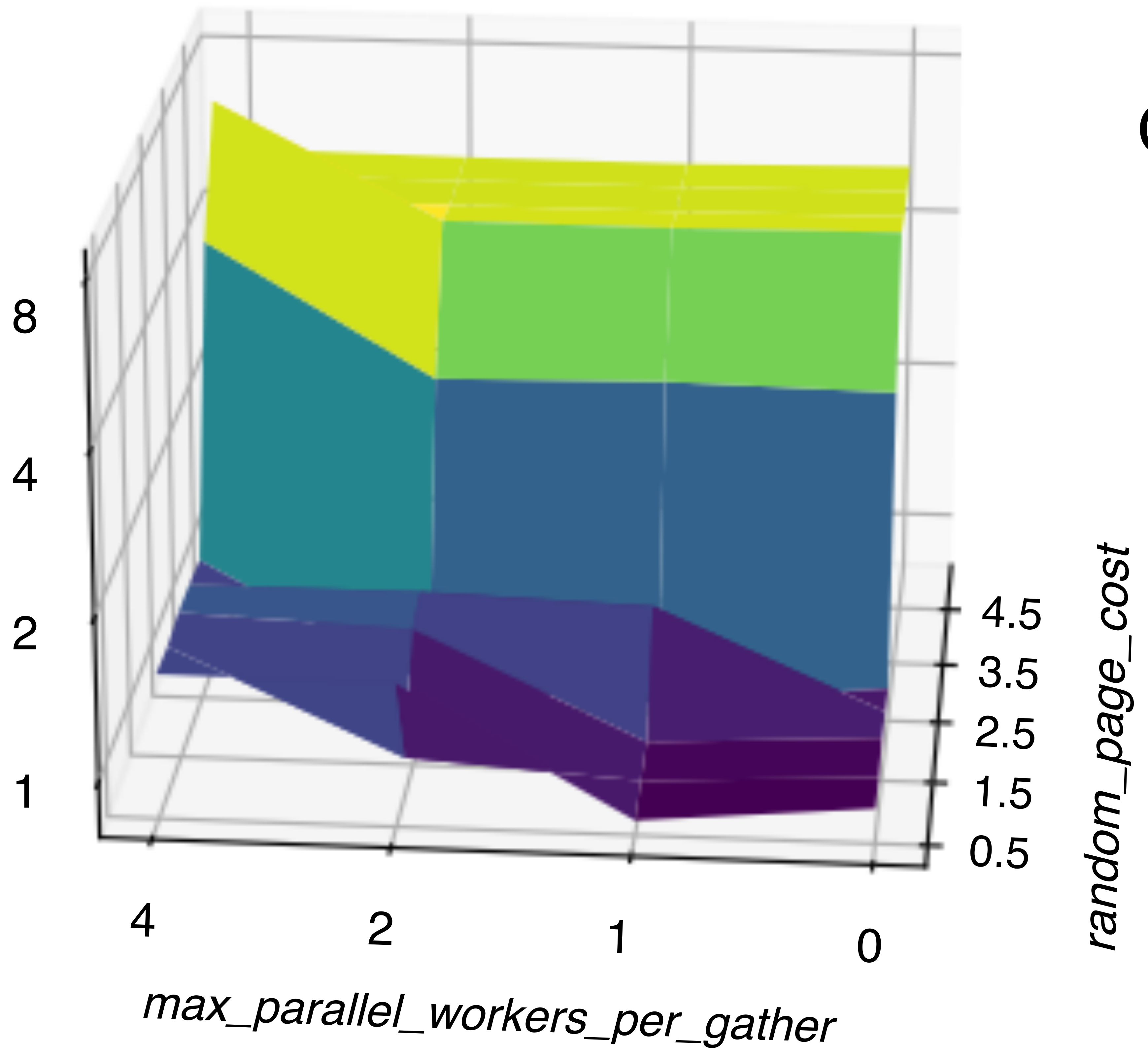
- ✓ Adjusting knobs to best fit the workload
- ✓ PostgreSQL parameters that are typically important: *work_mem*, *shared_buffers*, *max_wal_size*, etc.
- ✓ Example *max_parallel_workers_per_gather*: Max # of workers started by a Gather or Gather Merge node
- ✓ Example *random_page_cost*: Planner's cost of a non-sequentially fetched disk page
- ✓ These parameters highly depend on the application



Average query runtime tuning

for *max_parallel_workers_per_gather* and *random_page_cost*

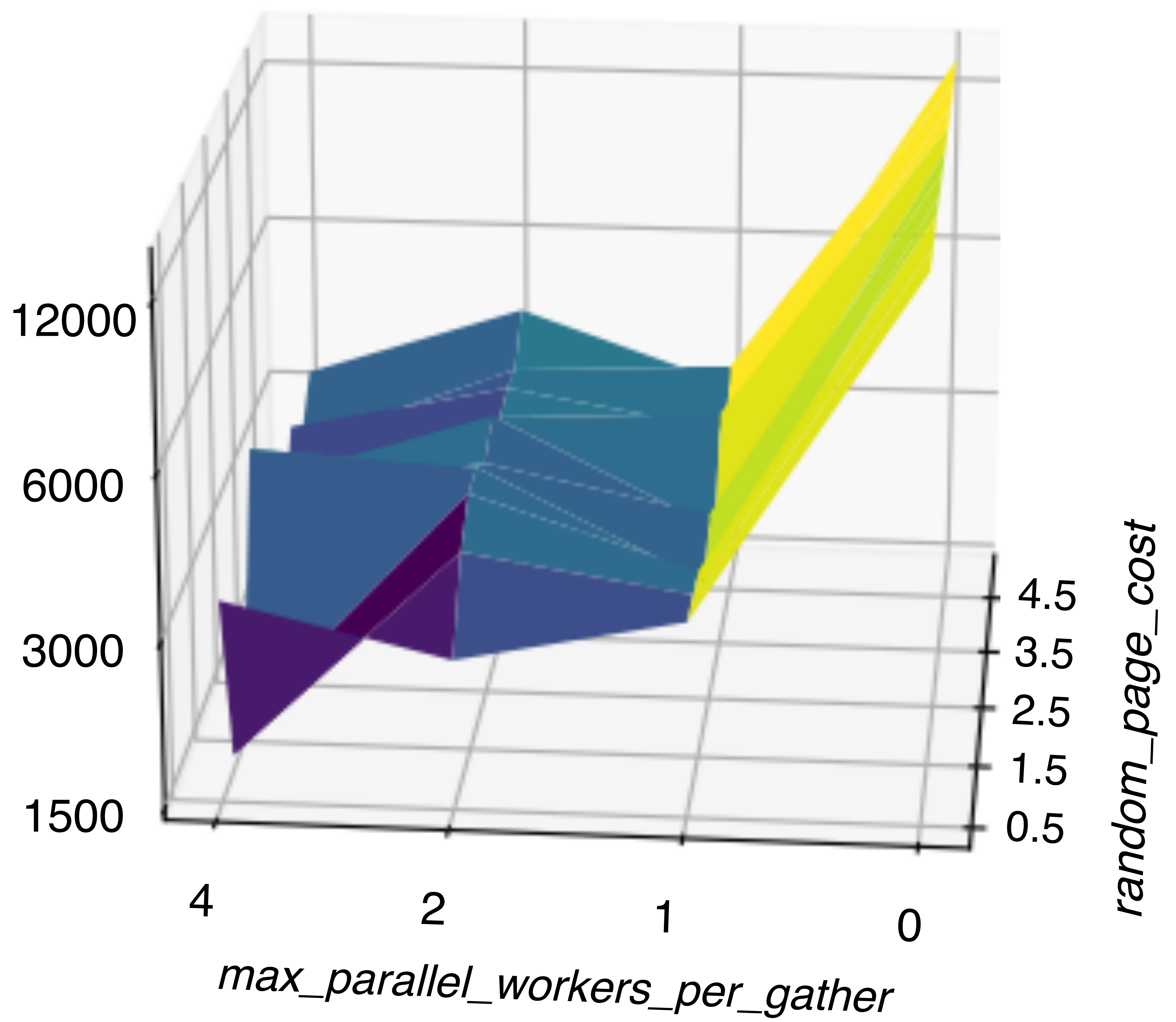
Epinions



Query runtime in ms

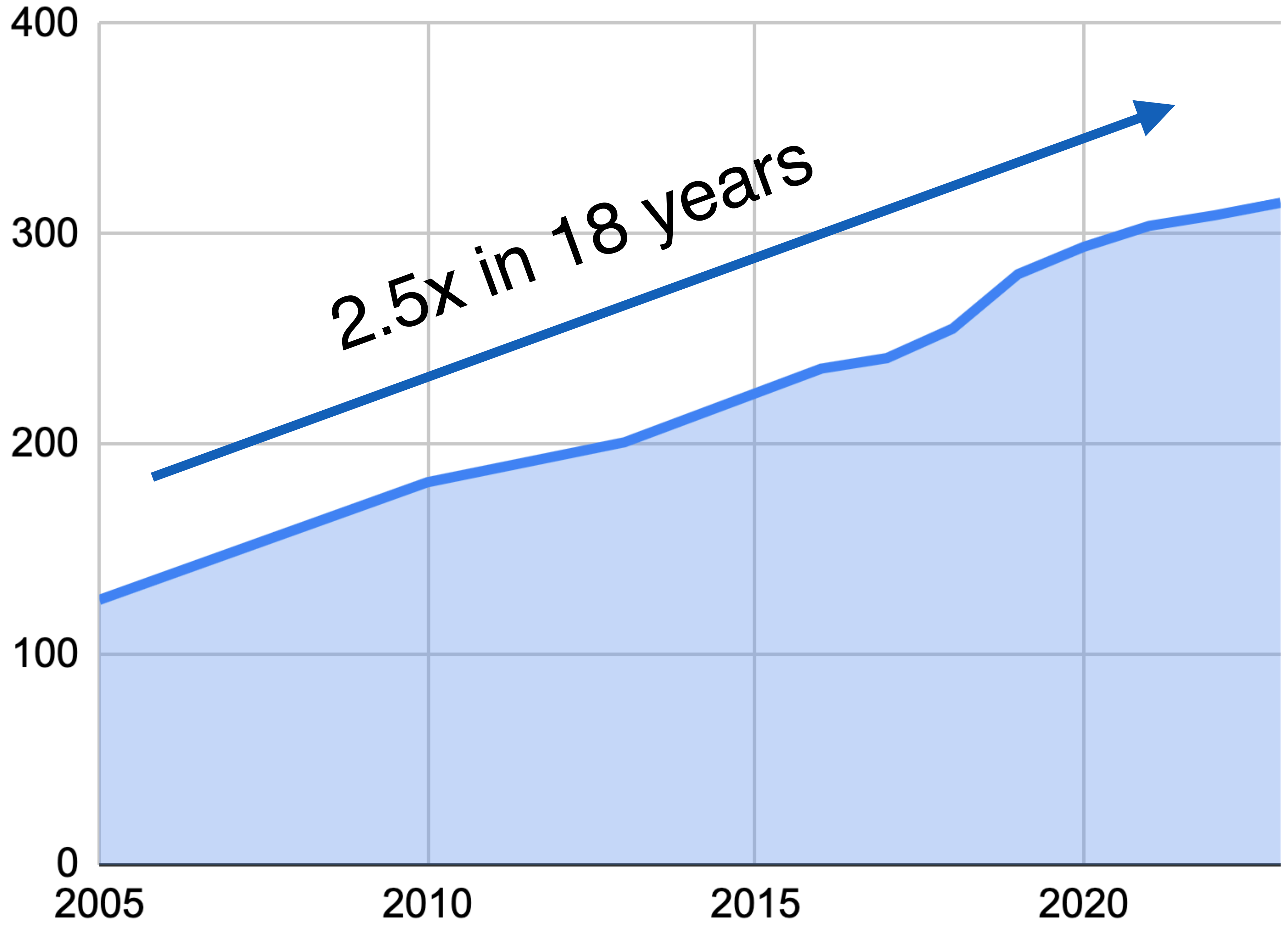
Lower the better

TPC-H



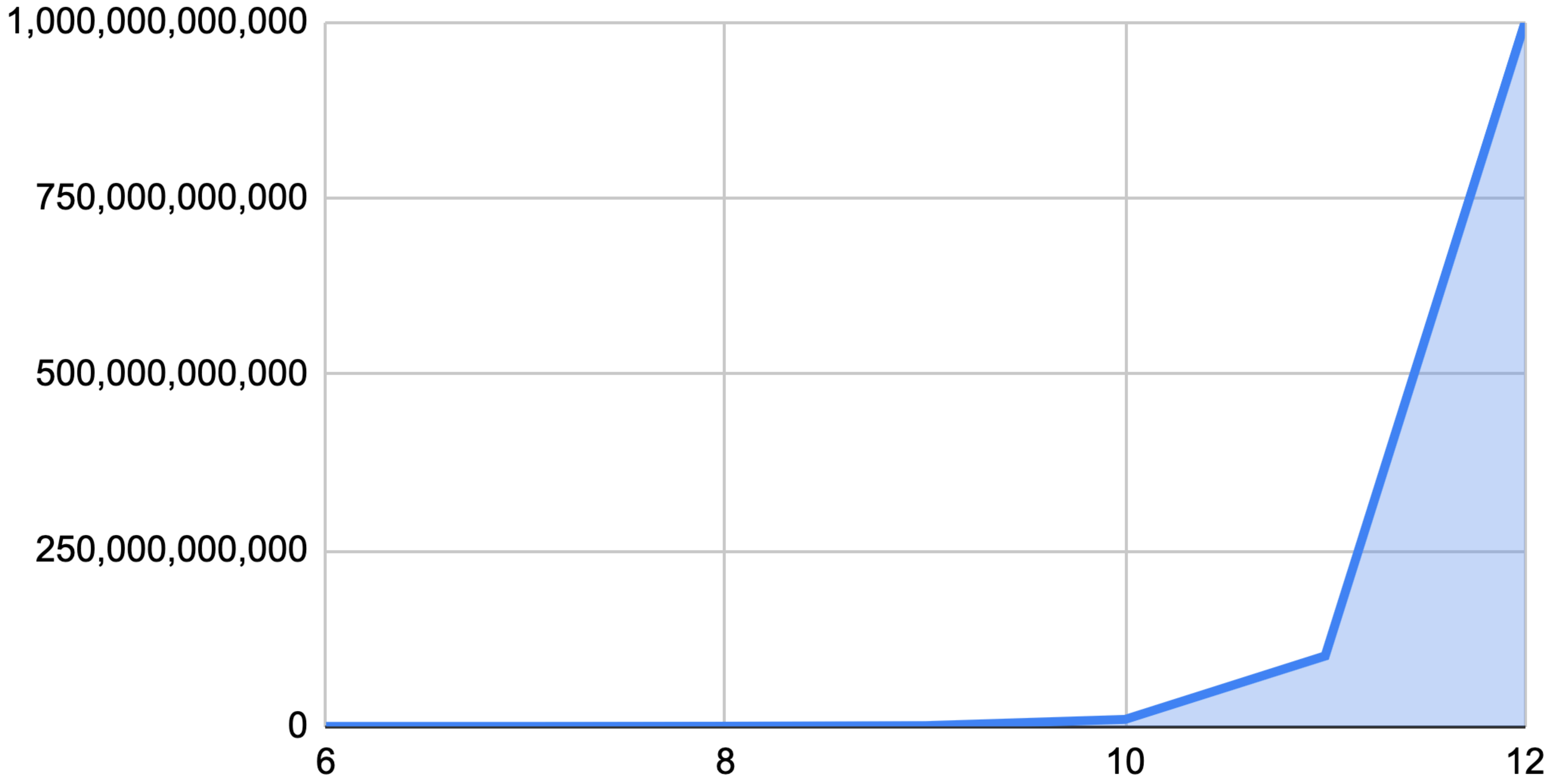
Complexity increasing over time makes performance even harder to attain

The number of parameters is growing **linearly**



PostgreSQL number of parameters

The number of configurations is growing **exponentially**



Example of complexity with 12 parameters

How is parameter tuning tackled today by DBAs and developers?

Manual

Tuning guru

Slow
Takes days

Painstaking
Needs high expertise

Ineffective
Tune again in a week

Inadequate
Seasonal workload

Heuristics

PGTune

POSTGRESQL CONFIGURATOR

One-size-fits-all
Uses generic rules

Workload agnostic
Not bespoke

Ineffective
Tune again in a week

Inadequate
Seasonal workload

dbtune

AI agent approach

A solution that learns by observation, adapts to changing workloads and autotunes with minimal supervision

Agentic AI for automated database tuning

PostgreSQL-specific AI/ML	›	DBtune learns how to solve PostgreSQL optimization challenges
Dynamic adaptation	›	Workload-specific
Easy to use	›	No need for background in AI or database tuning
Highly scaleable	›	Tunes a fleet in parallel no matter the complexity of each node

The ROI that users can expect using an agent to tune PostgreSQL

PostgreSQL spend

↓

Up to 50%[1]

Faster application

↑

Up to 10x[2]

Increase productivity

↑

Up to 25%

Reduction in CO2

↓

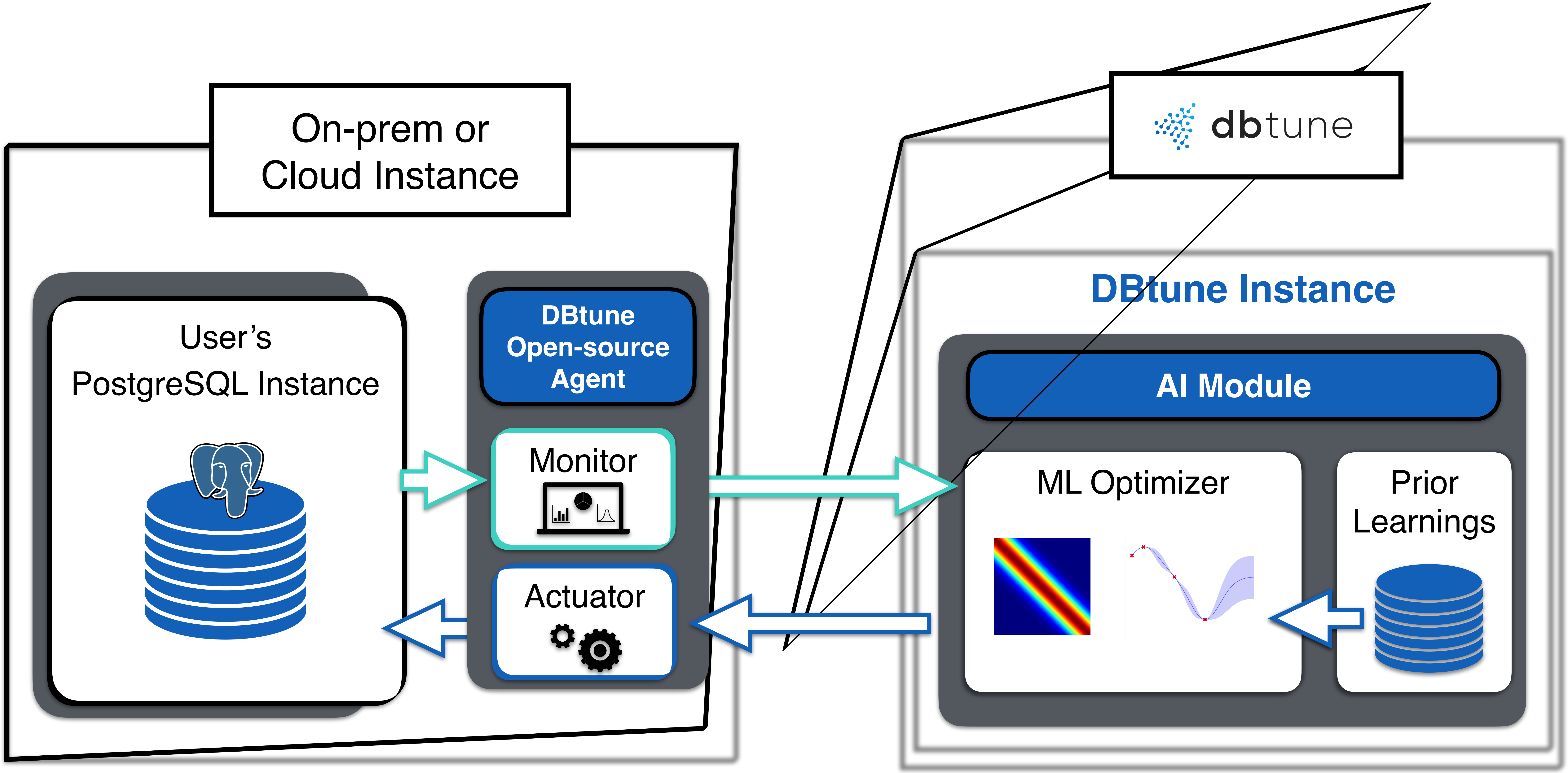
Up to 50%[3]

Agentic AI outcomes

1. https://dbtune.com/pdf/DBtune-deck-PGConf-EU.pdf
2. https://www.dbtune.com/blog/how-midwest-tape-achieved-a-10x-performance-boost-with-postgresql-tuning-on-aws-rds
3. https://www.datacenterdynamics.com/en/opinions/data-center-sustainability-is-no-longer-optional/

Agentic AI architecture for self-managed PostgreSQL

The DBtune use case



AI-driven performance for all PostgreSQL flavors

Community PostgreSQL

Amazon

RDS

Aurora

EDB EPAS

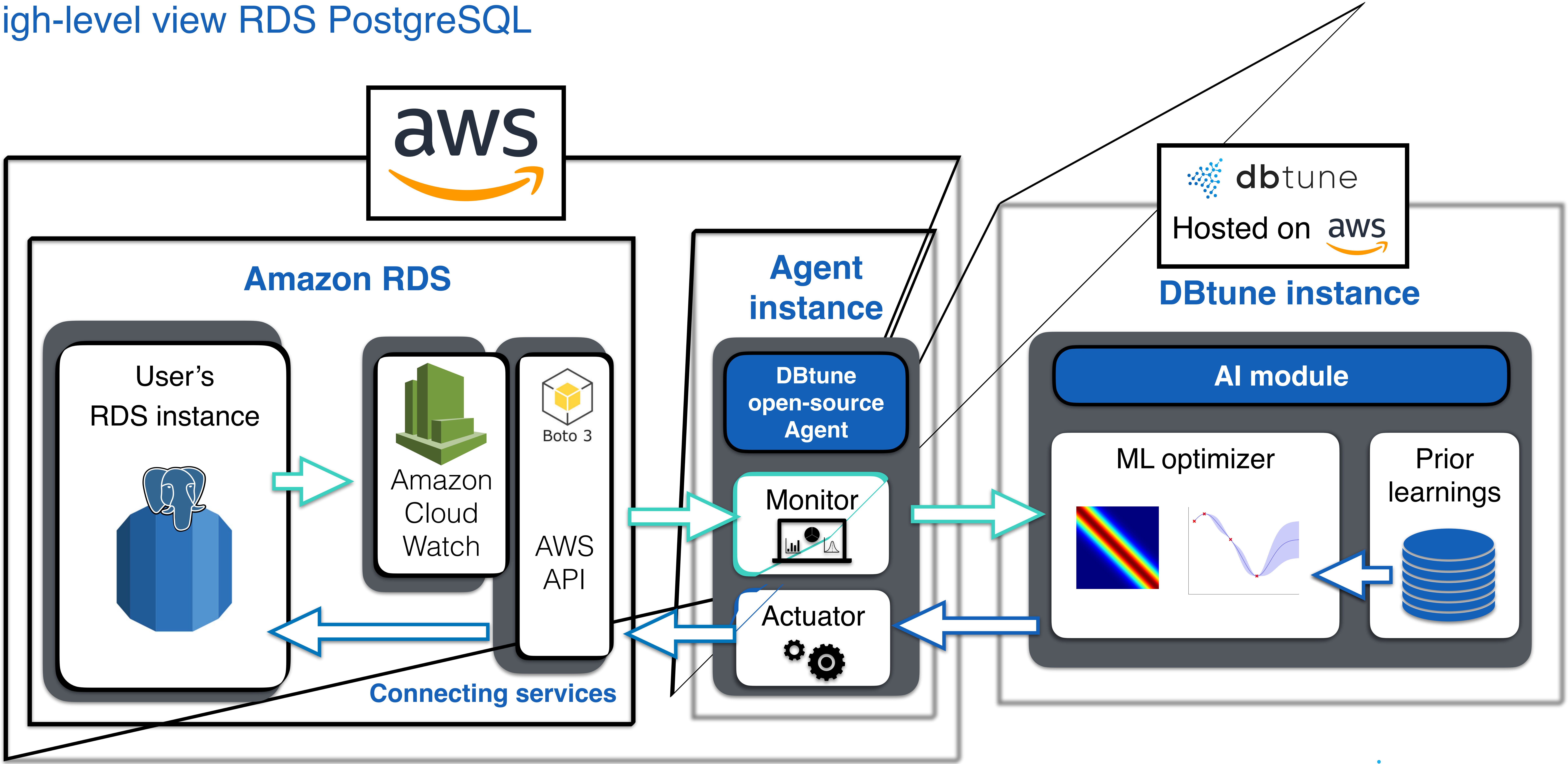
Aiven

Google Cloud SQL

Azure Flex Server

DBtune architecture for Database as a Service (DBaaS)

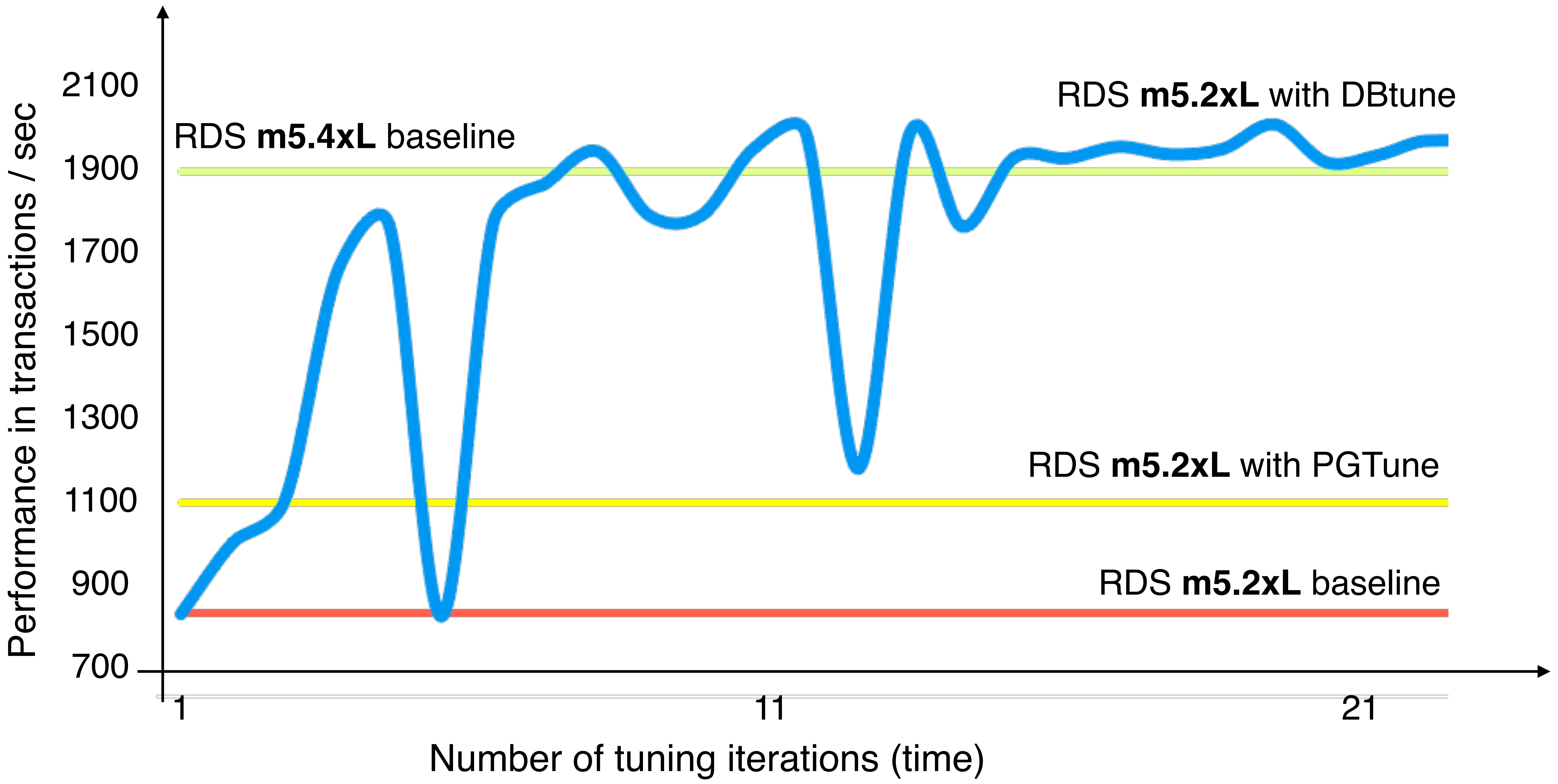
High-level view RDS PostgreSQL



Performance tuning example

DBtune doubles the performance of PostgreSQL Amazon RDS

Performance impact of tuning RDS m5.2xLarge cloud instance on the TPCC benchmark



DBtune on the smaller instance type achieves a level performance in excess of that achieved by an instance twice the size

< 3 hours

Proof of cost reduction

DBtune doubles the performance of PostgreSQL Amazon RDS

Hardware				Cost / Year		
AWS RDS Instance Type	Cores	RAM	IOPS	Instance	EBS	Total
db.m5.4xlarge	8	64 GBs	4000	12 475 US$	4 800 US$	17 275 US$
db.m5.2xlarge	4	32 GBs	2000	6 237 US$	2 400 US$	8 637 US$

Per instance savings: $8,638

- ✓ DBtune halves RDS cost (50% saving)
- ✓ Matches 4xLarge performance on a 2xLarge instance
- ✓ Medium and large companies use hundreds* of RDS instances

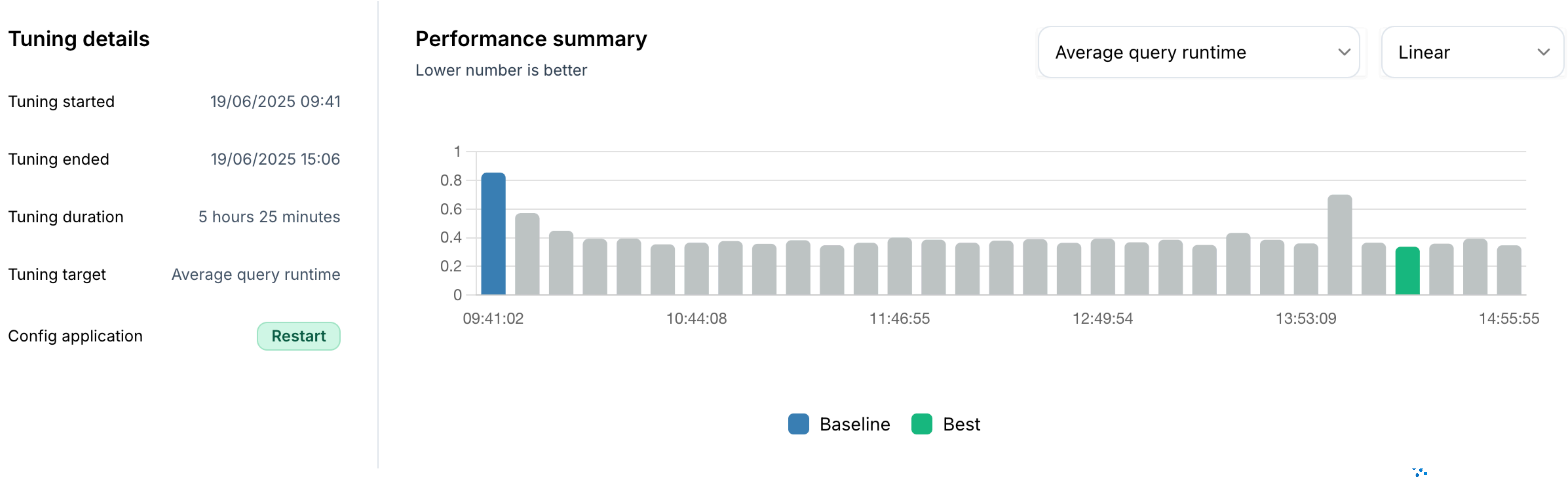
*A16z article: "The Cost of Cloud, a Trillion Dollar Paradox"

Insurance application use case study — Customer anonymized data

Environment: 16 vCPU, 32 GB RAM, on-prem, primary instance, PG 15

Manually tuned baseline by expert DBA

Automated tuning with DBtune



Tuning details

Tuning started	19/06/2025 09:41
Tuning ended	19/06/2025 15:06
Tuning duration	5 hours 25 minutes
Tuning target	Average query runtime
Config application	Restart

Performance summary

Lower number is better

Average query runtime ⌄

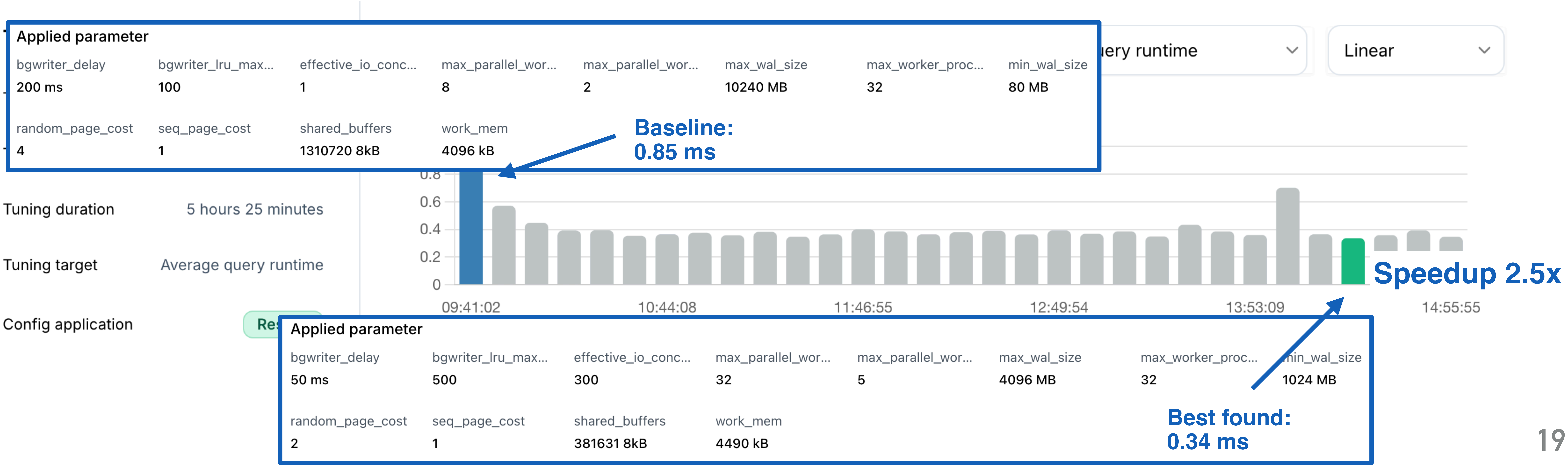
Linear ⌄

Insurance application use case study — Customer anonymized data

Environment: 16 vCPU, 32 GB RAM, on-prem, primary instance, PG 15

Manually tuned baseline by expert DBA

Automated tuning with DBtune



Applied parameter

bgwriter_delay	bgwriter_lru_max...	effective_io_conc...	max_parallel_wor...	max_parallel_wor...	max_wal_size	max_worker_proc...	min_wal_size
200 ms	100	1	8	2	10240 MB	32	80 MB
random_page_cost	seq_page_cost	shared_buffers	work_mem				
4	1	1310720 8kB	4096 kB				

Baseline:
0.85 ms

Tuning duration	5 hours 25 minutes
Tuning target	Average query runtime
Config application	Res

Applied parameter

bgwriter_delay	bgwriter_lru_max...	effective_io_conc...	max_parallel_wor...	max_parallel_wor...	max_wal_size	max_worker_proc...	min_wal_size
50 ms	500	300	32	5	4096 MB	32	1024 MB
random_page_cost	seq_page_cost	shared_buffers	work_mem				
2	1	381631 8kB	4490 kB				

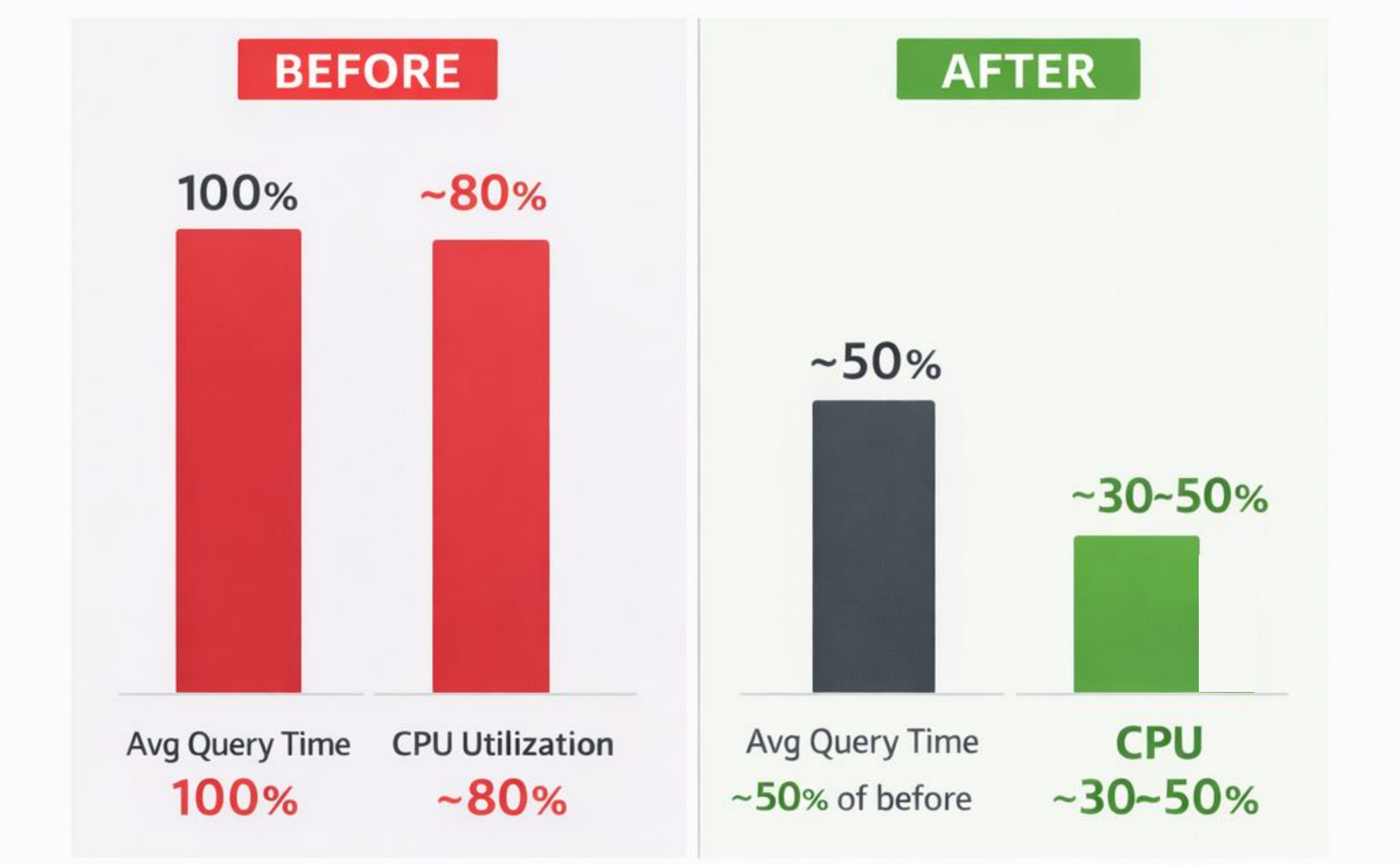
Best found:
0.34 ms

Speedup 2.5x

In production: Workforce management platform by Papershift

Environment: Amazon RDS m5.8xlarge, 32 vCPU, 128 GB RAM, PG 17.6

Baseline by RDS, automated tuning with DBtune

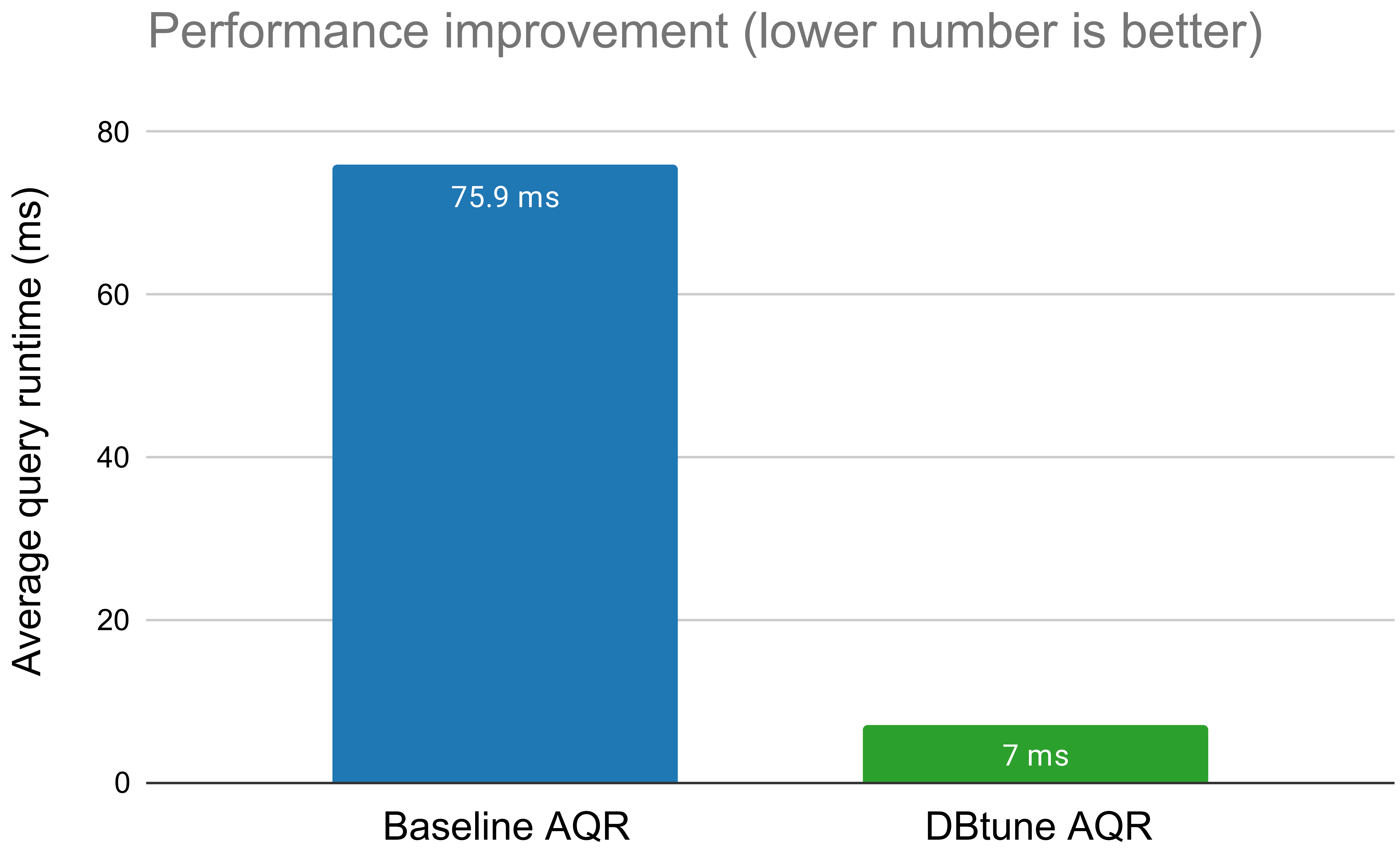


Blog: https://dev.to/floriansuchan/how-we-used-dbtune-to-cut-our-postgres-query-time-by-50-on-aws-rds-2a5e
Figure: https://www.linkedin.com/posts/vibhork_postgresql-autonomouspostgres-autonomousdatabase-activity-7408886284879835137-2i_m

In production: Digital content service by Midwest Tape

Environment: Amazon RDS r6g.12xlarge, 48 vCPU, 412 GB RAM, PG 14.17

Baseline by RDS, automated tuning with DBtune



Blog: https://www.dbtune.com/blog/how-midwest-tape-achieved-a-10x-performance-boost-with-postgresql-tuning-on-aws-rds

Safety in production environments

System guardrails to avoid unsafe configurations

Constrained optimization

Parameters have safe upper / lower limits in place

Memory monitoring guardrail

Real-time system memory monitoring to revert from potentially unsafe configurations

E.g. configuration that uses too much RAM — Triggered at 90% of RAM

Early exit condition

Optimization space may result in configuration with worse performance than default

This triggers early exit from existing configuration and move to next iteration

The future of database tuning is AI-assisted

Free edition: app.dbtune.com

nardiluigi

luigi@dbtune.com