PGECons
PostgreSQL Enterprise Consortium

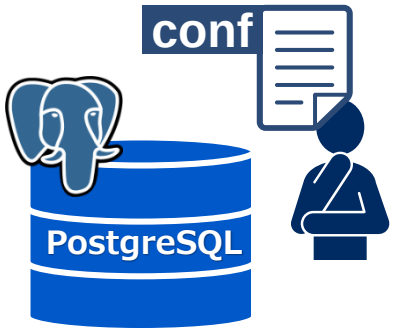
課題検討WG 2019年度活動報告

機械学習を用いた パラメータチューニングの自動化検証結果

PostgreSQL エンタープライズコンソーシアム
WG3 (課題検討WG)
株式会社富士通ソーシアルサイエンスラボラトリ 小山田 政紀

パラメータ自動チューニング班の活動目的

- テーマ選択にあたっての背景
  - PostgreSQLにはチューニング可能なパラメータが多く、最適な設定値の組み合わせを検討・検証する作業は、コストが掛かる。



- 目的
  - 機械学習の分野で利用されているソフトウェアやアルゴリズムをPostgreSQLパラメータチューニングに適用することで、性能改善を目的としたパラメータチューニングの自動化が可能か検証した。

機械学習におけるパラメータチューニング

- 予測精度の高いモデル開発のため、ハイパーパラメータの最適化が必要
  - ハイパーパラメータ：機械学習アルゴリズムの挙動を制御するために
    利用者がチューニング可能な（必要な）パラメータ

- 上記課題解決のため、ハイパーパラメータを自動で最適化する
  ソフトウェアやアルゴリズムが開発・公開されている。

- 本検証ではハイパーパラメータの最適化フレームワークとして人気がある
  「Optuna」を利用して、PostgreSQLのチューニングを実施した。

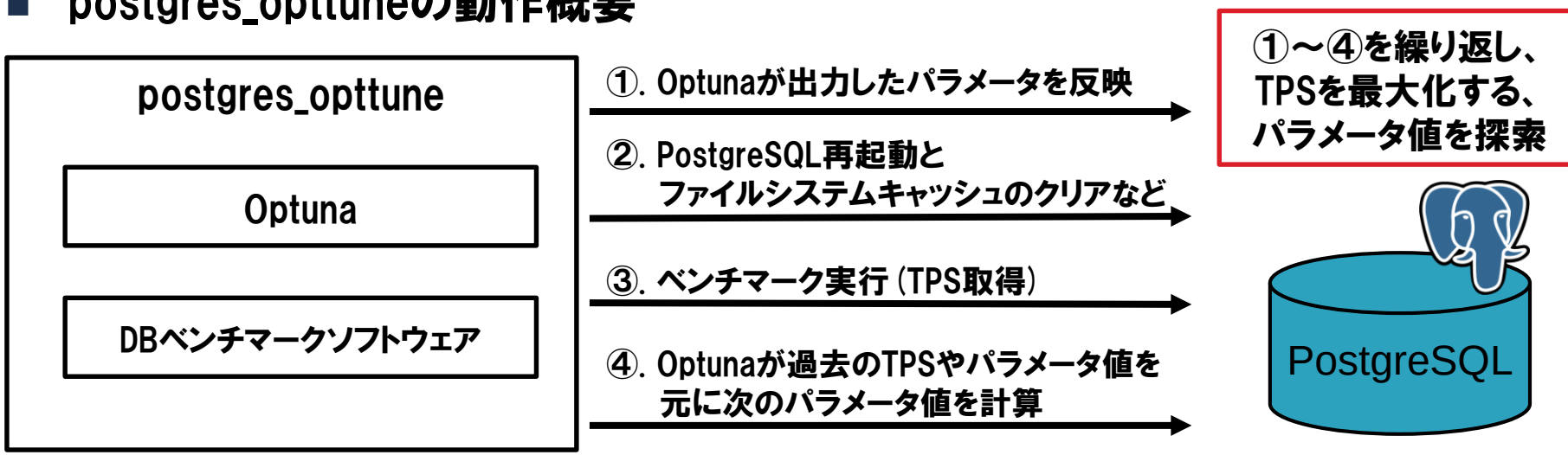
項番	諸元	概要
1	ソフトウェア名	Optuna
2	開発元	Preferred Network社
3	ライセンス	MIT License
4	開発言語	Python
5	GitHub	https://github.com/optuna/optuna

実験用ソフトウェア（postgres_opttune）

- Optunaを利用しPostgreSQLのパラメータ値を自動でチューニングする、実験用ソフトウェアを実装
  - ベンチマークソフトウェアのTPS（Transactions Per Second）を改善するためのパラメータ値を自動で探索

項番	諸元	概要
1	開発言語	Python
2	ライセンス	Apache-2.0 License
3	GitHub	https://github.com/ssl-oyamata/postgres_opttune

- postgres_opttuneの動作概要



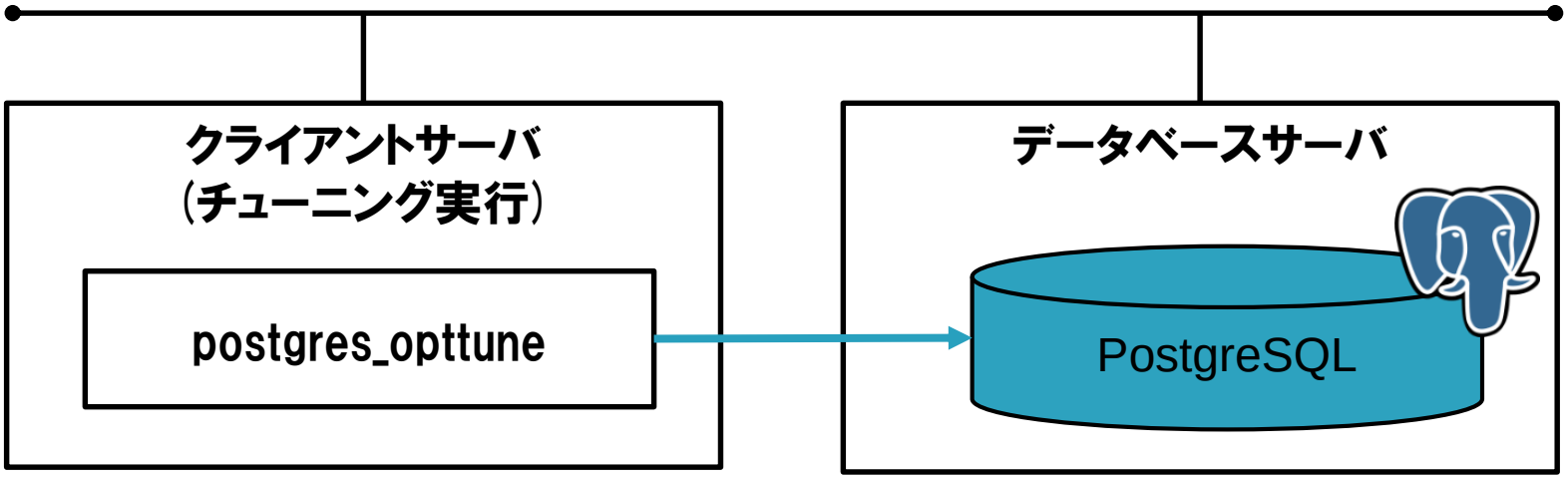
検証内容

- 検証内容①
  - 実験用ソフトウェア（postgres_opttune）のチューニング試行回数と、ベンチマークソフトウェアのTPSの関係を確認する。
    - 検証ではベンチマークソフトウェアとしてOltpbenchを利用
    - Oltpbenchに実装されたTPC-Cベンチマークで測定

項番	諸元	概要
1	開発言語	Java
2	ライセンス	Apache License, Version 2.0
3	GitHub	https://github.com/oltpbenchmark/oltpbench

- 検証内容②
  - 実験用ソフトウェア（postgres_opttune）のチューニング結果の評価のため、デフォルト値およびPGTuneとの比較を実施した。
  - PGTune：PostgreSQLが動作している環境（搭載メモリ、CPU数など）を元に、パラメータ推奨値を出力するWebサービス
    https://pgtune.leopard.in.ua/#/

検証環境



■ ハードウェア構成情報

名称	CPUコア数	メモリ(GiB)	ストレージの種類
データベースサーバ	4	8	SSD
クライアントサーバ	4	8	SSD

■ ソフトウェア構成情報

名称	バージョン	参照元
postgres_opttune	-	https://github.com/topics/postgres_opttune
optuna	1.0.0	https://github.com/optuna/optuna/
oltpbench	-	https://github.com/oltpbenchmark/oltpbench
PostgreSQL	12.1	https://www.postgresql.org/

検証条件（1/2）

- oltpbenchmark設定

項目	設定値	備考
ベンチマーク種別	tpcc	-
トランザクション分離レベル	TRANSACTION_READ_COMMITTED	-
スケールファクタ	50	約5GB
クライアント数	100	-
測定時間	1800s	測定時間は30min

- チューニング設定

項目	設定値	備考
最適化アルゴリズム	TPE （Tree-structured Parzen Estimator）（※1）	Optunaデフォルトの 最適化アルゴリズム
試行回数 （チューニング回数）	200	-

※1　ベイズ最適化（観測結果を基に現在よりも確率的に良い結果になると思われる、
パラメータを探索する手法）の一種。
TPEは結果のばらつきが少なく、コンスタントに最適なパラメータを探索できることが特徴。

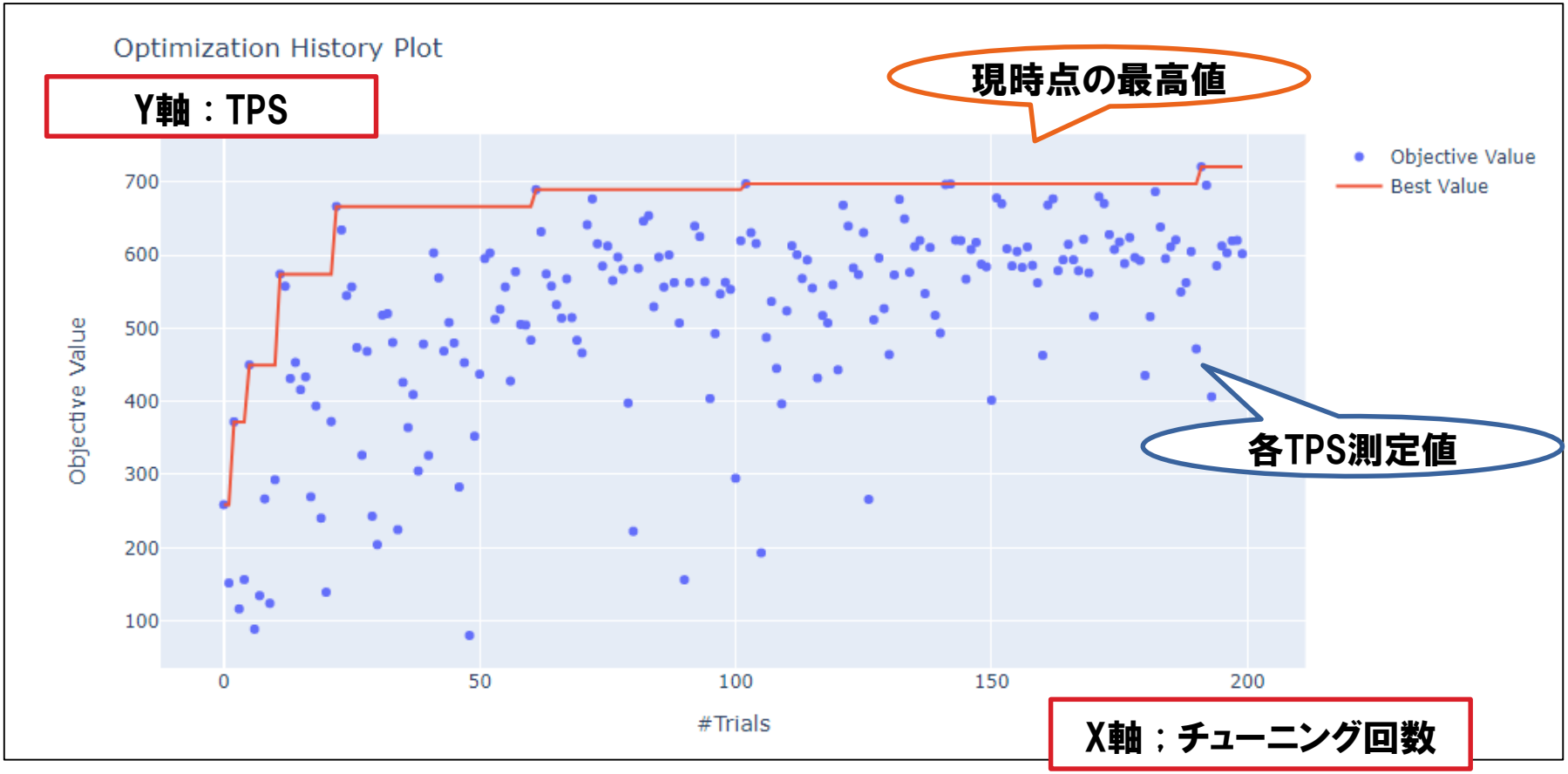
検証条件 (2/2)

- PostgreSQLのパラメータチューニング範囲

パラメータ	最小値	最大値	デフォルト値	データ型
bgwriter_lru_maxpages	0	1000	100	integer
checkpoint_completion_target	0.1	0.9	0.5	float
checkpoint_timeout	1min	30min	5min	time
default_statistics_target	100	2048	100	integer
effective_cache_size	4GB	8GB	4GB	bytes
effective_io_concurrency	1	1000	1	integer
max_parallel_maintenance_workers	0	8	2	integer
max_parallel_workers_per_gather	0	8	2	integer
max_wal_size	256MB	16GB	1GB	bytes
random_page_cost	1	10	4.0	float
shared_buffers	128MB	6GB	128MB	bytes
temp_buffers	8MB	1GB	8MB	bytes
wal_buffers	1MB	256MB	16MB	bytes
wal_compression	on, off		off	boolean
wal_writer_delay	10ms	10s	200ms	time
work_mem	4MB	1GB	4MB	bytes

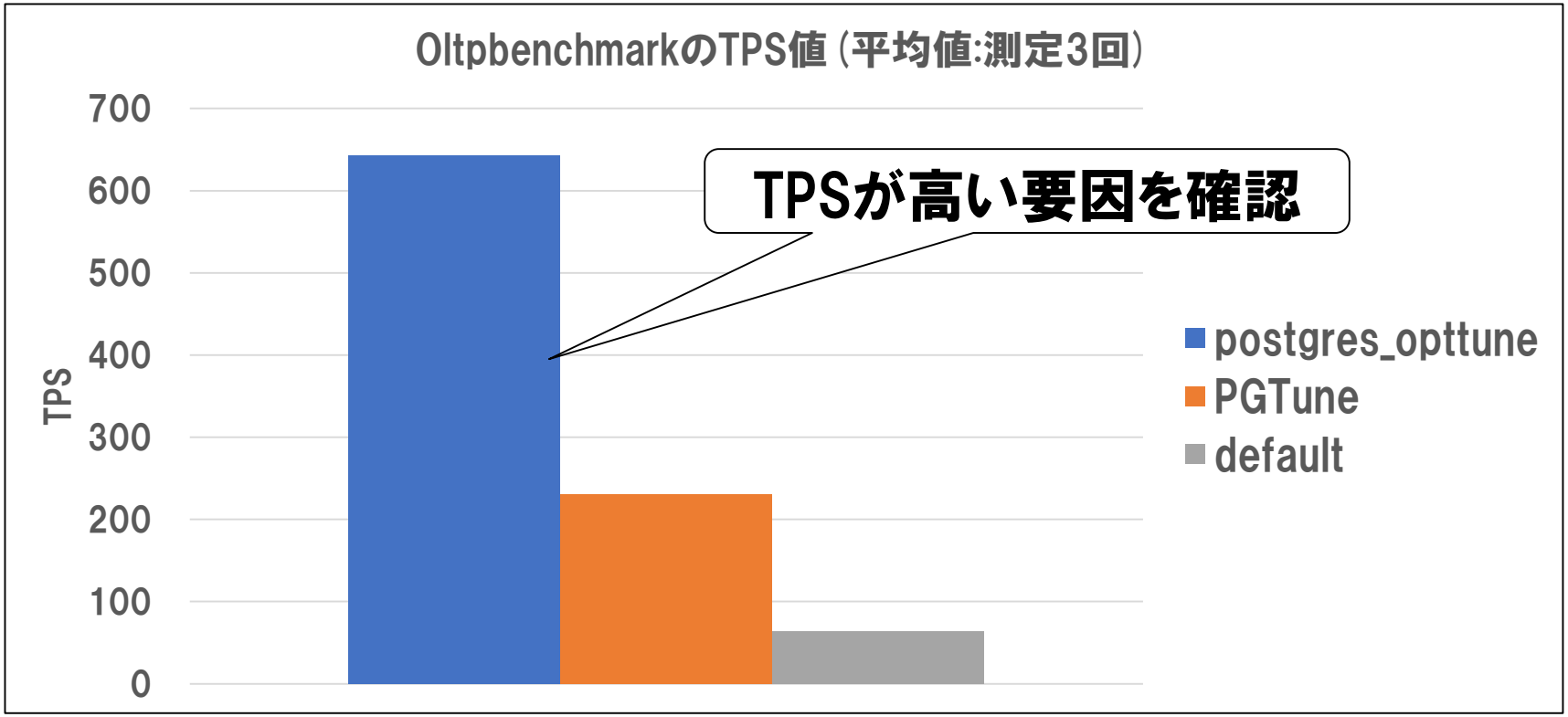
検証結果①：チューニング回数とTPS確認結果

- チューニング回数を重ねる毎にTPSが改善
  - チューニングが100回を超えた辺りから劇的な改善はない



検証結果②：デフォルト値などとの比較

- 上述の検証①でTPSが最も高い値となったパラメータ値を、postgres_opttuneで最適化されたパラメータとして利用
- oltpbenchをそれぞれの条件で3回実行し、平均値を算出
- 想定以上に本手法のTPS値が高い
  - 要因については次頁に記載



TPS改善の要因調査（パラメータ比較）

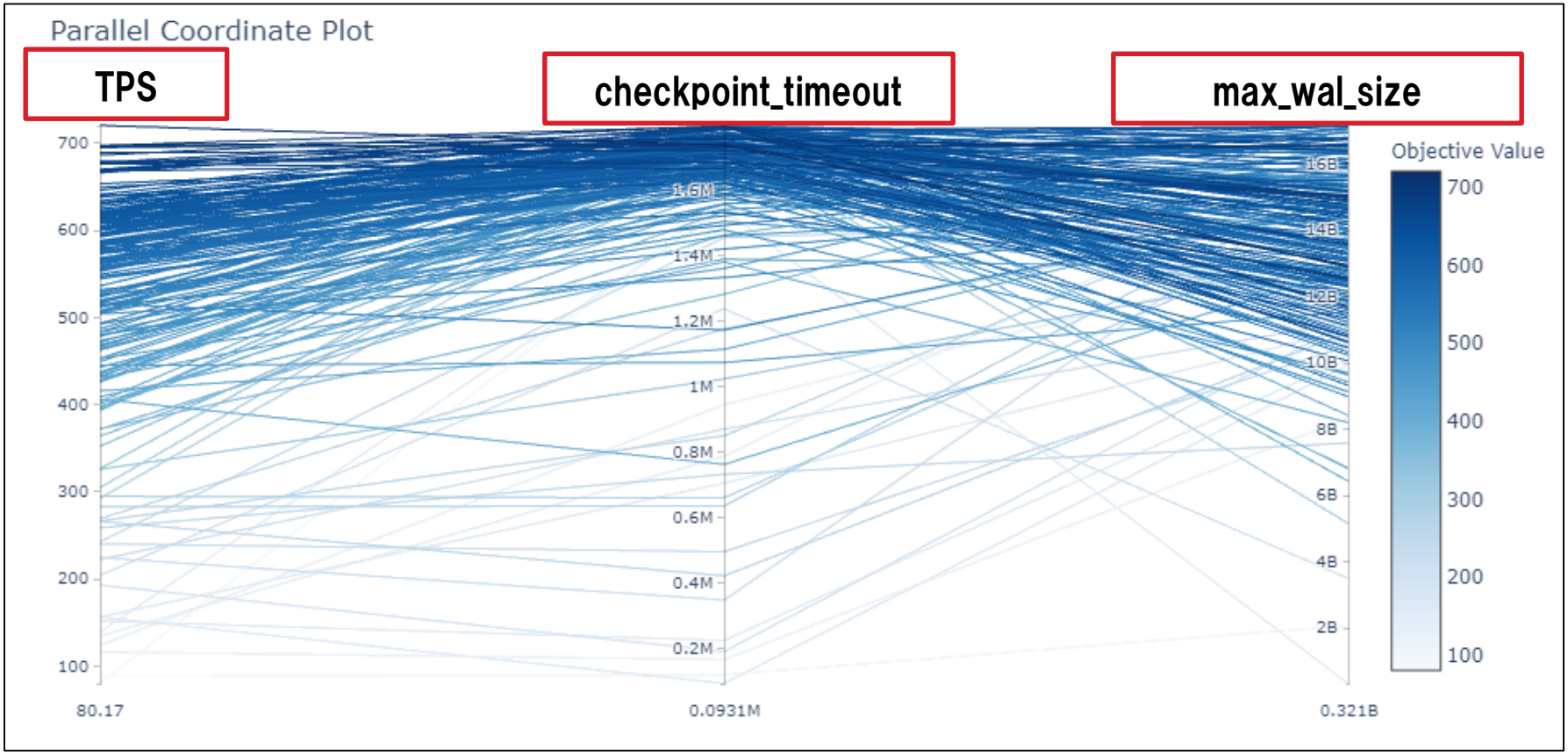
- ベンチマーク実行中に性能低下を招く、
  チェックポイント処理を実施しないようにようにチューニング
  - 上記チューニングの場合、PostgreSQLクラッシュ時のリカバリに時間がかかる

パラメータ	postgres_opttune	PGTune（参考）	default（参考）
bgwriter_lru_maxpages	469	(100)	(100)
checkpoint_completion_target	0.658923793689129	0.9	(0.5)
checkpoint_timeout	1740s（29min）	(5min)	(5min)
default_statistics_target	100	100	(100)
effective_cache_size	6623MB	6GB	(4GB)
effective_io_concurrency	613	200	(1)
maintenance_work_mem	(64MB)	512MB	(64MB)
max_parallel_workers	4	4	(8)
max_parallel_workers_per_gather	5	2	(2)
max_wal_size	12359MB（12.07GB）	4GB	(1GB)
max_worker_processes	(8)	4	(8)
min_wal_size	(80MB)		
random_page_cost	7.477068166087535		
shared_buffers	5216MB		
temp_buffers	689MB		
wal_buffers	95MB		
wal_compression	on		
wal_writer_delay	7s		
work_mem	766MB		

チェックポイント処理回数：
postgres_opttune=0回
PGTune=5回
default=6回

チェックポイント関連パラメータの性能影響確認

- 線の色が濃い程、TPSが高いことを示しており、checkpoint_timeoutおよびmax_wal_sizeを増やすことで、TPSが向上していることを確認



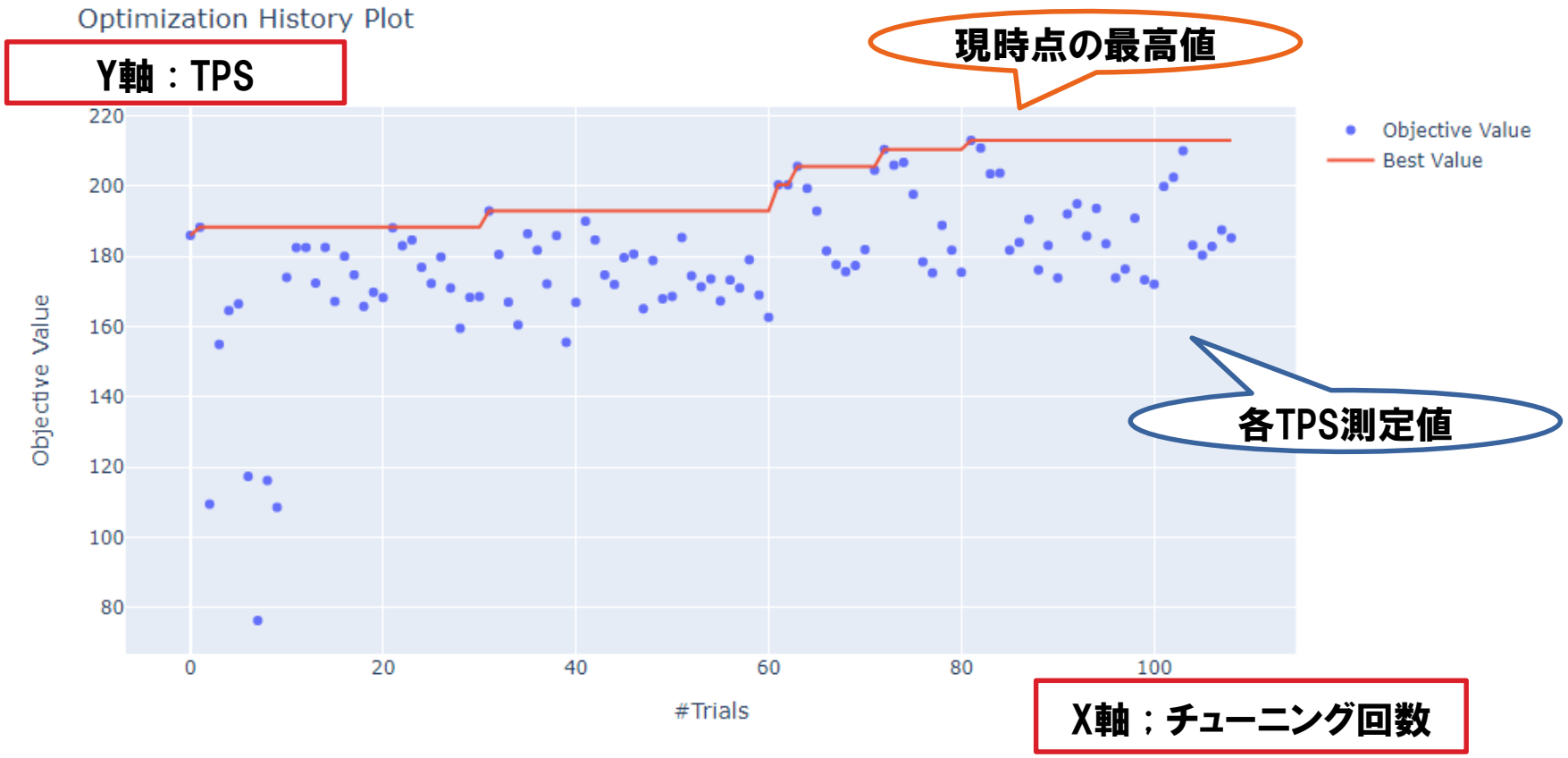
追加検証

- チェックポイント関連のパラメータ値のチューニング結果
  - TPSを最大化するため、性能低下を招く、
    チェックポイント処理を実施しないようにようにチューニング

- 追加検証
  - チェックポイント関連のパラメータを除外した場合の、
    チューニング結果について評価

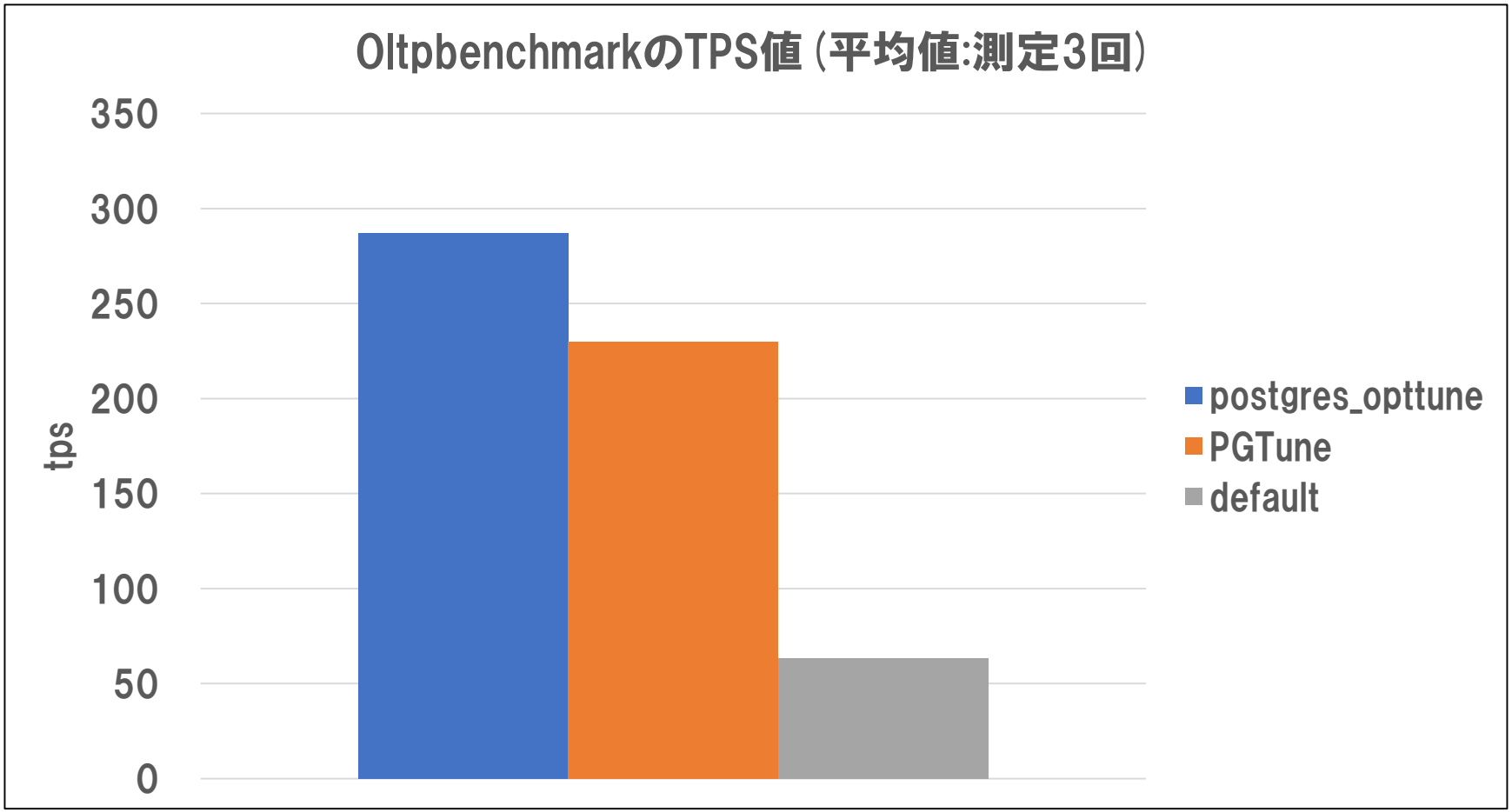
追加検証結果①：チェックポイント関連パラメータ除外

- チェックポイント関連のパラメータを除外しチューニングを実施
  - checkpoint_timeout = 5min, max_wal_size = 4GBで固定 (PGtuneと同じ設定)
  - チューニング回数は100回



追加検証結果②：チェックポイント関連パラメータ除外

- 上述の検証と同じく、TPSが最も高かったパラメータを設定
- 性能差は小さくなったが、他の手法よりTPSが高い



まとめ

- Optunaに実装されたパラメータの最適化アルゴリズムをPostgreSQLに適用することで、特定のベンチマークにおけるTPSの改善を確認
  - 最適化する処理が決まっていれば、自動チューニングは可能

- 本測定条件ではチェックポイント関連のパラメータの値を増やした影響により、TPSが大幅に改善
  - ただし、チェックポイント関連のパラメータは、PostgreSQLクラッシュ時のリカバリ時間に影響するため、クラッシュしてから復旧するまでの許容時間などを考慮し、決定する実装が必要

課題と対応方針

- 最適化するワークロードのサンプリング
  - PostgreSQLの稼働統計情報やログファイルを元に、
    該当の時間に実行されたトランザクション群をサンプリング
  - サンプリングしたトランザクション群を最適化する仕組みの検討・実装

- チェックポイント関連パラメータの自動チューニング検討
  - チェックポイント関連パラメータの値と、
    PostgreSQLクラッシュ時のリカバリ時間の関係を調査
  - PostgreSQLのクラッシュ時の復旧目標時間を元に、
    チェックポイント関連パラメータの値をチューニングする仕組みを検討

- チューニング回数（試行回数）を減らす施策の検討

パラメータチューニングにかかる作業負荷の軽減を目指して頑張ります！！

PGECons

PostgreSQL Enterprise Consortium