

# 分散SQL処理系に対する問合せ実行時間予測モデル のベンチマーク構築

安 柏勳<sup>1,a)</sup> 椎名峻平<sup>2,b)</sup> 中森辰洋<sup>1,c)</sup> 川島英之<sup>1,d)</sup>

## 1. 背景

近年、データの爆発的な増加とクラウド基盤の普及により、データベースシステムにおける問合せ処理は一層大規模化・複雑化している。これに伴い、問合せの実行時間を予測することは、ユーザーのストレス軽減、リソース割当てやスケジューリング、クラウド課金コストの最適化などにおいて極めて重要な役割を果たしている。

従来は、コストベース最適化 (Cost-Based Optimizer: CBO) における推定値やヒューリスティックに基づく手法が広く用いられてきたが、複雑な問合せや不正確な統計情報に起因して、十分な精度を得られない場合が多いという課題が指摘されている。この問題に対処するため、近年では機械学習を用いた実行時間予測手法が提案されている。これらの手法は、問合せ実行計画の特徴量やオペレータ単位の情報を入力とし、問合せやデータセットの特徴などを学習して高い推定精度を実現している。

## 2. 研究課題

既存の実行時間予測手法は数多く提案されているが、その多くは PostgreSQL のような単一ノード上で動作する逐次実行型データベースを対象としている。一方で、Trino [1] や Apache SparkSQL [2] のようにノード間通信や並列実行を伴う分散 SQL エンジンに特有の特性は十分に考慮されていない。分散環境では、ネットワーク遅延、スケジューラの挙動、タスク間の負荷分散など、実行時間に影響を及ぼす要因が多岐にわたる。そのため、従来のモデルをそのまま適用しても精度が低下すると考えられる。したがって、

分散並列環境に特化した実行時間予測モデルの体系的な評価と比較分析が求められる。

## 3. 関連研究

Flat Vector [3] アプローチは、実行計画の木構造を平坦化し、ベクトルとして表現する手法である。各オペレータのタイプ、推定コスト、カーディナリティといった情報をベクトルにマッピングするベースラインモデルである。

Zero-shot [4] はインスタンス固有の特徴量を使わず、演算子・述語・テーブルをグラフ構造で表現し、GNN を用いることで高い予測精度を達成している。

DACE [5] は Transformer のエンコーダーを用い、コストモデルの推定値と木構造エンコーディングを利用して軽量かつ高精度な実行時間予測を実現する。

これらの手法は逐次実行型データベース上で高い精度を達成しているが、Trino のようなノード間通信や並列実行を伴う分散 SQL エンジンに特有の特性 (ネットワーク遅延、スケジューラの挙動、タスク間の負荷分散など) を十分に考慮しておらず、そのまま適用しても精度が低下する可能性がある。また、分散環境では問合せ実行計画のサブ計画 (ステージやタスク) の正確な実行時間を取得することが困難であり、訓練時にそれらの情報を利用する DACE のようなモデルは精度が低下する可能性が懸念される。

## 4. Trino

Trino は、分散環境において高速かつスケーラブルに SQL クエリを実行するために設計された分散 SQL エンジンである。ストレージを内部に持たず、外部ストレージ上のデータに対して直接アクセスを行う「クエリエンジン型」の設計を採用している。このアーキテクチャにより、複数のデータソースを統合的に扱うフェデレーテッドクエリ処理が可能となり、大規模データ分析やクラウド環境における ETL 処理などで広く利用されている。類似のシステム

<sup>1</sup> 慶應義塾大学

<sup>2</sup> トヨタ自動車株式会社

a) anpohsun13211@keio.jp

b) shumpei.shiina@mail.toyota.co.jp

c) tatsuhironm@keio.jp

d) river@sfc.keio.ac.jp

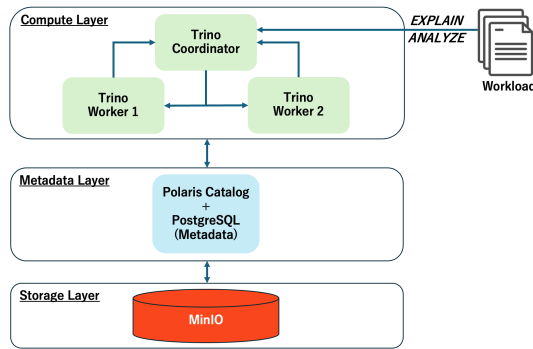


図 1: 問合せ実行環境

として、Apache SparkSQL などが挙げられる。

本研究では、Trino を代表的な分散 SQL 処理系の一例として採用し、問合せ実行時間予測モデルのベンチマークを構築した。

## 5. 実験設計

### 5.1 問合せ実行環境

訓練データである問合せ実行計画を取得するため、Trino を中心に、Iceberg テーブル形式および Polaris Catalog、MinIO を組み合わせた環境を構築し、既存手法の学習に用いる問合せ実行計画を収集した (図 1)。

Trino は、問合せ解析および実行計画の生成を担う Coordinator ノードと、実際に問合せを並列実行する Worker ノード群で構成される。Coordinator は問合せを解析し、オペレータ単位に分割されたタスクを各 Worker にスケジューリングする。Polaris Catalog は、Iceberg テーブルのメタデータを管理するコンポーネントであり、テーブルスキーマ、スナップショット履歴、マニフェストファイルのパスなどを PostgreSQL に永続化する。Trino はクエリ実行時に Polaris に対して REST API 経由でメタデータを問い合わせ、対象データの位置やスナップショットなどを取得する。MinIO は、AWS S3 と互換性をもつオブジェクトストレージとして機能し、Iceberg テーブルを構成する Parquet ファイルやマニフェストファイルを格納する。Trino の各 Worker は Polaris から得たファイルパスを基に、MinIO 上のデータに直接アクセスして分散並列に読み込みを行う。

以上の構成により、実際の分散 SQL エンジンにおける問合せ処理を再現し、ノード間通信・スケジューリング・ストレージ I/O など、分散実行特有の要素を含めた問合せ実行時間予測モデルの評価が可能となる。

### 5.2 評価方法

評価においては、既存の実行時間予測手法を Trino 環境向けに再実装し、分散 SQL エンジン上での適用可能性を評価した。再実装には、Heinrich らの公開コードを基に、Trino 向けに拡張を行った [6]。

評価に用いるデータセットは、Zero-shot [4] の研究で提

案されたベンチマークを使用した。これらの問合せを本研究の Trino 環境 (セクション 5.1) で実行し、取得した問合せ実行計画と実測実行時間を用いて各モデルの学習を行った。推定精度の指標として Q-Error を用いた評価を行った。

$$Q\text{-Error} = \max \left( \frac{\text{prediction}}{\text{actual}}, \frac{\text{actual}}{\text{prediction}} \right) \quad (1)$$

Q-Error は、予測値と実測値の比率に基づく評価指標であり、実行時間予測のようにスケールの異なるクエリを含む場合でも相対的な誤差を適切に評価できる利点を持つ。

## 参考文献

- [1] Trino Software Foundation. *Trino: Distributed SQL Engine*. Available at: <https://trino.io/>. Accessed: October 2025.
- [2] Michael Armbrust et al. 2015. *Spark SQL: Relational Data Processing in Spark*. In Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data (SIGMOD '15), Association for Computing Machinery, New York, NY, USA, 1383–1394. <https://doi.org/10.1145/2723372.2742797>
- [3] A. Ganapathi et al. *Predicting Multiple Metrics for Queries: Better Decisions Enabled by Machine Learning*. 2009 IEEE 25th International Conference on Data Engineering (ICDE'09), Shanghai, China, 2009, pp. 592–603. <https://doi.org/10.1109/ICDE.2009.130>
- [4] Benjamin Hilprecht et al. 2022. *Zero-shot cost models for out-of-the-box learned cost prediction*. Proc. VLDB Endow. 15, 11 (July 2022), 2361–2374. <https://doi.org/10.14778/3551793.3551799>
- [5] Z. Liang et al. *DACE: A Database-Agnostic Cost Estimator*. 2024 IEEE 40th International Conference on Data Engineering (ICDE'24), Utrecht, Netherlands, 2024, pp. 4925–4937. <https://doi.org/10.1109/ICDE60146.2024.00374>
- [6] Roman Heinrich et al. 2025. *How Good are Learned Cost Models, Really? Insights from Query Optimization Tasks*. Proc. ACM Manag. Data 3, 3, Article 172 (June 2025), 27 pages. <https://doi.org/10.1145/3725309>