社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

データベースシステムにおける 問合せ最適化器の評価に関する実験的考察

川道　亮治† 　早水　悠登† 　合田　和生† 　喜連川　優†,††

† 東京大学生産技術研究所　〒 153–0041 東京都目黒区駒場 4–6–1
†† 国立情報学研究所　〒 101–8430 東京都千代田区一ツ橋 2–1–2
E-mail: †{kawamichi,haya,kgoda,kitsure}@tkl.iis.u-tokyo.ac.jp

あらまし　コストベースの問合せ最適化器において，コスト見積りの正確性はデータベースシステムの性能に極めて大きな影響を与えるにも関わらず，その定量的評価を系統的に行う手法は十分に確立されているとはいえない．本論文では，コスト見積りの正確性の定量評価指標を提案し，実験的にその有効性を検討する．
キーワード　データベースシステム, 問合せ最適化, コストベース最適化, コストモデル

An Experimental Study on Evaluating Query Optimizers in Database Systems

Ryoji KAWAMICHI†, Yuto HAYAMIZU†, Kazuo GODA†, and Masaru KITSUREGAWA†,††

† Institute of Industrial Science, The University of Tokyo, Komaba 4–6–1, Meguro-ku, Tokyo, 153–0041 Japan
†† National Institute of Informatics, Hitotsubashi 2–1–2, Chiyoda-ku, Tokyo, 101–8430 Japan
E-mail: †{kawamichi,haya,kgoda,kitsure}@tkl.iis.u-tokyo.ac.jp

Abstract　For a cost-based query optimizer, even though the accuracy of cost estimation has a significant impact on performance of a database system, quantitative evaluation methods of the accuracy have not been sufficiently studied yet. In this paper, we propose evaluation indices of the accuracy of cost estimation, and discuss their effectiveness with experimental evaluations.
Key words　database system, query optimization, cost-based optimization, cost model

1. はじめに

データベースシステムの実効的な性能を考える上で，問合せ最適化は最も重要な技術といってよく，多様かつ数多くの取り組みが行われてきたが [1]．今日において実用的とされるデータベースシステムの殆どは，System-R [2] に端を発する**コストベース最適化**の枠組みを採用している．即ち，問合せ実行計画を構成するオペレータに対してコストモデルを定め，コストモデルにより算出されるコスト見積値が最小となる問合せ実行計画を探索する．初期の System-R においては，ストレージマネージャに対するシステムコール回数をコストとする単純なコストモデルが用いられていたが，CPU インテンシブな問合せ実行計画と入出力インテンシブな問合せ実行計画の比較や，あるいはネットワーク入出力量や利用可能帯域の異なる問合せ実行計画など，データベース環境の多様化に伴って考案された様々な実行方式の優劣を判断するため，現代的な問合せ最適化器においては，種々の単位オペレーションに要するコストを定めたうえで，その組み合わせによりコストモデルを構成することが一般的となっている．

コストを測る尺度は，問合せ実行計画の最適性の定義により異なるが，実行時間が用いられる場合が多い．この場合，理想的なコストモデルによるコスト見積値は実行時間と一致するが，現実の実装においてはテーブルのカーディナリティ推定誤差や，コストモデルにおける近似等によりコスト見積り誤差が生じる．特定のアクセスメソッドや結合方式の組み合わせにおいて誤差の程度が著しい場合も少なくなく，その場合にはコストが問合せ実行計画の優劣を比較する尺度として有効に機能せず，問合せ最適化器の精度が大きく損なわれる可能性がある．

問合せ最適化の精度は，データベースシステムの実用性に直結することから，問合せ最適化器の品質改善，あるいは品質保証に関して様々な取り組みが行われてきている．特に問合せ最適化器による問合せ実行計画選択精度に関しては，問合せ実行計画のランク付けの精度による評価手法や [3]，問合せ最適化の正答が既知であるような検証用データベース及びクエリ生成手法 [4] などが見られる．これら手法では，問合せ最適化器によって結果的に選択される問合せ実行計画が正しいか否かに着目しているが，問合せ実行計画の精度を向上させるという観点からすると，コストモデルによるコスト見積値の尺度としての

表 1　PostgreSQL コストモデルにおける単位オペレーションと単位コストパラメータ
Table 1　Unit operations and their cost parameters in cost models of PostgreSQL

	単位オペレーション	単位コストパラメータ
o_1	1 ページのシーケンシャル I/O	`seq_page_cost`
o_2	1 ページのランダム I/O	`random_page_cost`
o_3	基本演算子 1 回の実行	`cpu_operator_cost`
o_4	1 タプルを出力	`cpu_tuple_cost`
o_5	索引ページ上の 1 タプル走査	`cpu_index_tuple_cost`

正確性を評価する定量的指標が必要であると考えられる．

本論文では，コストモデルによるコスト見積りの誤差，ならびに異なる問合せ実行計画間でのコスト見積りの整合性を定量化するための指標を提案し，問合せ最適化器の評価におけるその有効性を実験的に検討する．コスト見積りの誤差の指標としては，コスト見積値の絶対誤差，相対誤差にそれぞれ着目した指標を定め，またコスト見積りの整合性の指標としては，問合せの選択率空間におけるコスト見積値および問合せ実行時間の連続性に着目した指標を定め，これらの算出方法を提示する．実験においては，TPC-H ベンチマークとオープンソースデータベースシステム PostgreSQL を利用し，複数の TPC-H の問合せに対して当該指標を算出することで，その有効性を議論する．

本論文の構成は次の通りである．2 節では，コストベース最適化の枠組み，ならびに本論文で提案する評価指標について説明する．3 節では，TPC-H クエリを用いた実験により評価指標の有効性を明らかにする．4 節では関連研究をまとめ，最後に 5 節にて全体をまとめる．

2. コストモデルとその評価

2.1 コストモデルによるコスト見積り

コストベース最適化において，問合せ実行計画の優劣を判断するためのコストは目的に応じて適宜定義すればよいが，最終的に 1 つの問合せ実行計画を選択するためにコストの見積値は全順序集合に属することが望ましく，またデータベースシステムにおいては多くの場合問合せ実行時間が短いことが良しとされるため，コスト見積値として予測される実行時間の定数倍を出力するコストモデルが採用される場合が多い．例えば，オープンソースデータベースシステムである PostgreSQL においては，問合せ処理は表 1 に示す 5 種類の単位オペレーションのシーケンスにより構成されると仮定し，各単位オペレーション o_i の単位コストパラメータ c_i と o_i が実行される回数 n_i を用いて，問合せ実行計画 P のコスト $C(P)$ を次のように定義するコストモデルの枠組みを採用している．

$$C(P) = c_1 n_1 + c_2 n_2 + \cdots + c_5 n_5 = \mathbf{c} \cdot \mathbf{n}$$
$$(\mathbf{c} \equiv (c_1, \cdots, c_5), \mathbf{n} \equiv (n_1, \cdots, n_5))$$

単位コストパラメータは事前設定により与えられ，PostgreSQL の提供する各アクセスパスや結合方式に対して $\mathbf{n}$ を算出するコストモデルが定められている．例えば選択条件の一切ない全表走査については

$$\mathbf{n}_{\mathrm{seqscan}} = ((\text{表の全ページ数}), 0, 0, (\text{表の全タプル数}), 0)$$

と比較的単純なコストモデルとなる．一方で，索引走査に関しては B^+ 木の高さに基づく索引ページ取得数や，Mackert らの手法 [5] に基づく索引ページおよび表ページのキャッシュヒット率の予測，索引キーによる表ページ走査のシーケンシャル性の判定などの手続きにより $\mathbf{n}$ が算出されるほか，ハッシュ結合においてはハイブリッドハッシュ結合における入出力の有無の判定，利用可能メモリ量やタプル長に基づくハッシュバケット数・ハッシュ衝突確立の予測などに基づき $\mathbf{n}$ が算出されるなど，比較的複雑な手続きの組み合わせによってコストモデルが構成されている．

2.2 コストモデル誤差・整合性評価指標

先に述べた PostgreSQL の例に示されるように，現代的なコストベース最適化におけるコストモデルは複雑な手続きの組み合わせにより実現されており，また少なからずその計算方法は経験的知識に基づく近似によって構成されているため，任意の問合せ実行計画に対して正確なコスト見積りを実現することは難しい．また，問合せ実行方式の改善や新規追加に応じたコストモデルの変化が，最終的なコスト見積値やプラン選択に与える影響の見積りは自明ではない．

本論文では，コスト見積り値と実際の実行時間の測定値から算出される指標を定義し，当該指標によってコストモデルの誤差および整合性を定量的に把握可能であるかを実験的に考察する．当該指標の有用性が示されれば，例えばコストモデル開発において典型的なクエリに対するコストモデルの有効性を確認する指針として利用できるほか，ユーザが自ら利用しているデータベースシステムが問合せ実行計画の選択を誤りやすいパターンを確認する，といった用途において活用できることが期待される．

2.2.1 評価指標算出のためのサンプル点抽出

コスト見積り値と実行時間との誤差を評価するために，選択率 σ の評価用問合せ $Q(\sigma)$ に対して，複数の σ の値 $\sigma_1, \sigma_2, \cdots, \sigma_N$ において，問合せ最適化器の選択する問合せ実行計画 $p(\sigma_i)$，ならびに $p(\sigma_i)$ のコスト見積値 $C(p(\sigma_i))$ を記録するとともに，実行時間 $T(\sigma_i)$ を測定し，サンプル点とする．σ_i の取り方は一意ではないが，本論文においては，選択率のオーダーに対してサンプル点が均等に分布するよう，下記の方法を採用した．

$$\sigma_1 = \sigma_{\min}, \sigma_2 = \sigma_{\min} \times \alpha, \cdots, \sigma_N = \sigma_{\min} \times \alpha^{N-1}$$
$$\alpha \equiv \left(\frac{\sigma_{\max}}{\sigma_{\min}}\right)^{\frac{1}{N-1}}$$

上記の方法によりサンプル点を取ると，選択条件における選択率 σ が異なる以外は全く同一の問合せ実行計画の対 $p(\sigma_i), p(\sigma_j)(i \neq j)$ が複数生じることが予想される．このような実行計画は問合せ実行計画パターン P に属する $(p(\sigma_i) \in P, p(\sigma_j) \in P)$ と定義する．ある問合せ実行計画パターン P に属する全ての問合せ実行計画は，問合せ実行計画を構成する演算子の組み合わせが全て同一であり，同一の計算手

順によりコスト見積りが行われるため，$p(\sigma) \in P$ なる全ての σ のサンプル点によって，問合せ実行計画パターン P のコストモデルの誤差・整合性の評価を行うことができる．以降の議論においては，問合せ $Q(\sigma)$ に対して得られたサンプル点において観測された問合せ実行計画パターンを $P_1, P_2, \cdots$ と表記し，また問合せ実行計画パターン P_i が得られる σ の全ての値の集合を $\hat{\sigma}_i$ と表記する．

2.2.2 誤差指標 E_{RMSE}: コスト見積値と実行時間の二乗平均平方根誤差

コスト見積値の誤差として，実際の実行時間との差の絶対値は最も自明な指標である．サンプル点 σ における当該誤差を次のように表記する．

$$e_{\mathrm{AD}}(\sigma) = |T(\sigma) - C(p(\sigma))| \tag{1}$$

これを用いて，各問合せ実行計画パターン P_i について，コスト見積値と実行時間の二乗平均平方根誤差を算出し，誤差指標 $E_{\mathrm{RMSE}}(P_i)$ と定義する．

$$E_{\mathrm{RMSE}}(P_i) = \sqrt{\frac{1}{|\hat{\sigma}_i|} \sum_{\sigma \in \hat{\sigma}_i} e_{\mathrm{AD}}(\sigma)^2} \tag{2}$$

2.2.3 誤差指標 E_Q: コスト見積値と実行時間の Q-error 平均

2 つ目の誤差指標として，ここでは Q-error [6] を用いた誤差評価を行う．Q-error とは，実測値が真値から相対的にどれだけ乖離しているかを表す値であり，具体的には次の式で表される．

$$e_Q(\sigma) = \begin{cases} \frac{T(\sigma)}{C(p(\sigma))} & (T(\sigma) \geqq C(p(\sigma))) \\ \frac{C(p(\sigma))}{T(\sigma)} & (C(p(\sigma)) \geqq T(\sigma)) \\ \infty & (C(p(\sigma)) = 0 \vee T(\sigma) = 0) \end{cases} \tag{3}$$

この場合，コスト見積値が実行時間の 10 倍の場合と，1/10 の場合いずれも $e_Q(\sigma) = 10$ となる．これにより，実行時間の絶対値で測る場合には誤差が小さい場合にも，相対的には大幅に誤差が生じているような状況を捉えることが可能となると期待される．Q-error を用いた誤差指標 E_Q は，次のように定義する．

$$E_Q(P_i) = \frac{1}{|\hat{\sigma}_i|} \sum_{\sigma \in \hat{\sigma}_i} e_Q(\sigma) \tag{4}$$

2.2.4 整合性指標 $I \pm s$: 問合せ実行計画パターン損益分岐点における実行時間の非連続性

問合せ実行計画パターン P_i に属し，選択率 σ である問合せ実行計画を $P_i(\sigma)$ と表記する．複数の問合せ実行計画パターン $P_1, P_2, \cdots$ が観測されるクエリにおいては，問合せ最適化器により選択される問合せ実行計画のパターンが P_i, P_j の間で切り替わる損益分岐点 $\sigma_{\mathrm{BEP}ij}$ が存在し，$P_i(\sigma_{\mathrm{BEP}ij}) \approx P_j(\sigma_{\mathrm{BEP}ij})$ である(注1)．P_i に属するサンプル点から外挿された $\sigma = \mathrm{BEP}ij$ のときの T の値を $T'(\sigma_{\mathrm{BEP}ij}|P_i)$，$P_j$ に関しても同様に $T'(\sigma_{\mathrm{BEP}ij}|P_j)$ とすると，P_i, P_j 間の整合性が保たれている場合には $T'(\sigma_{\mathrm{BEP}ij}|P_i) \approx T'(\sigma_{\mathrm{BEP}ij}|P_j)$ である．しかし，コスト見積誤差により両者の値に乖離が生じる場合には，選択率 $\sigma_{\mathrm{BEP}ij}$ 周辺では急峻な実行時間の変動が生じる．即ち，特に $\sigma_{\mathrm{BEP}ij}$ 周辺において P_i, P_j のコスト見積値が整合性のある尺度として機能せず，コストの大小を有効に比較できないことを意味する．このような整合性を表す指標として，$T'(\sigma_{\mathrm{BEP}ij}|P_i), T'(\sigma_{\mathrm{BEP}ij}|P_j)$ の差の絶対値の平均値を I，標準偏差を s として整合性指標 $I \pm s$ を定義する．

$$\delta T'(\sigma_{\mathrm{BEP}ij}) = \left| T'(\sigma_{\mathrm{BEP}ij}|P_i) - T'(\sigma_{\mathrm{BEP}ij}|P_j) \right| \tag{5}$$

$$I = \frac{1}{|\hat{\sigma}_{\mathrm{BEP}}|} \sum_{\sigma \in \hat{\sigma}_{\mathrm{BEP}}} \delta T'(\sigma_{\mathrm{BEP}ij}) \tag{6}$$

$$s = \sqrt{\frac{1}{|\hat{\sigma}_{\mathrm{BEP}}|} \sum_{\sigma \in \hat{\sigma}_{\mathrm{BEP}}} (\delta T'(\sigma_{\mathrm{BEP}ij}) - I)} \tag{7}$$

(注1)：選択率およびコスト見積値は連続値であるとは限らないため，等値ではない可能性もある．

表 2　実験環境諸元
Table 2　The experimental environment

Dell PowerEdge R720xd server	
Processor	Intel Xeon E5-2680 (2 sockets, 16 cores)
Memory	64GB (x8 8GB 1,600MHz DDR3 DIMM)
Storage	RAID6(22D+2P) w/ x24 NL-SAS 10Krpm HDDs
OS	CentOS 5.10 (Linux 2.6.18)

表 3　較正対象パラメータとその設定値
Table 3　Calibrated parameters and obtained values

Parameter	Calibrated value
`seq_page_cost`	1.34×10^{-5}
`random_page_cost`	2.68×10^{-3}
`cpu_operator_cost`	3.08×10^{-8}
`cpu_tuple_cost`	1.05×10^{-7}
`cpu_index_tuple_cost`	2.98×10^{-7}

3. 評価実験

3.1 実験環境

本論文に用いた実験環境の諸元を表 2 に示す．当該環境において PostgreSQL 9.2.17 を利用し，TPC-H ベンチマークのデータセット (scale factor = 10) を生成し，データロードすることで実験用データベースを作成した．

実験に先立ち，PostgreSQL の見積もるコストが問合せ実行時間と同じ尺度となるよう，[7] の手法により問合せ最適化器のパラメータ較正を行った．較正対象としたパラメータと，較正により得られた数値を表 3 に示す．較正においては，各単位オペレーションの実行回数が自明な異なる 5 つの較正用問合せ $q_1, \cdots, q_5$ を用意し，それぞれの $\mathbf{n}$ を $\mathbf{n_1}, \cdots, \mathbf{n_5}$，また較正の実施環境におけるそれぞれの実行時間の測定値を $\mathbf{t} = (t_1, \cdots, t_5)$ とすると，下記の五元一次連立方程式が得られる．

$$\mathbf{Nc} = \mathbf{t}$$
$$(\mathbf{N} \equiv (\mathbf{n_1}, \mathbf{n_2}, \mathbf{n_3}, \mathbf{n_4}, \mathbf{n_5})^{\mathrm{T}})$$

```
SELECT sum(l_quantity)
  FROM customer, orders, lineitem
  WHERE c_custkey <= [selection-key]
    AND c_custkey = o_custkey
    AND l_orderkey = o_orderkey;
```

図 1　簡略化した TPC-H Q.3
Fig. 1　Simplified TPC-H Q.3

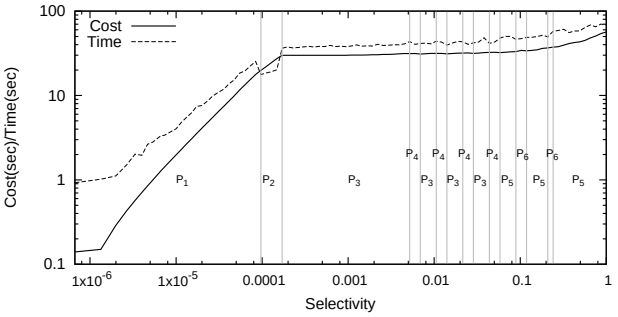

図 2　Q.3 のコストおよび実行時間
Fig. 2　Estimated cost and execution time of Q.3

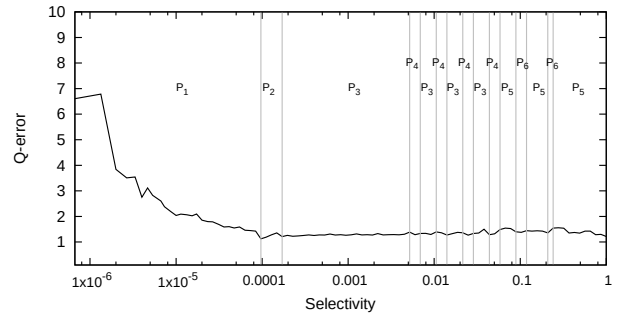

図 3　Q.3 の実行時間とコスト見積値の Q-error
Fig. 3　Q-error between execution time and estimated cost of Q.3

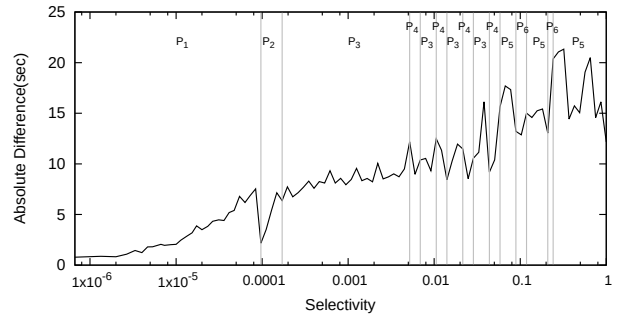

図 4　Q.3 のコストと実行時間の差の絶対値
Fig. 4　Absolute value of the difference between cost and execution time of Q.3

N が正定値行列となるよう較正用問合せを設計することで，上記連立方程式の解として各パラメータの較正値が得られる．

3.2　TPC-H Q.3 を用いた評価

まず，図 1 に示す簡略化した TPC-H Q.3 を用いて，問合せのコスト見積りと実際の実行時間を比較する実験を行った．当該問合せでは，customer 表の選択条件を調整することにより，問合せの選択率を 0%から 100%まで変化させることが可能である．本実験では，2.2.1 で述べたとおり，選択率のオーダーに対してサンプル点が 101 個均等に分布するようにした．すなわち

$$\sigma_{\min} = \frac{100}{M}(\%)$$
$$\alpha = M^{\frac{1}{N-1}}$$
$$N = 101$$

となるように選択条件を変化させた（選択率 0%は対数化ができないため含めていない）．ここで M は customer 表の c_custkey の最大値であり，scale factor=10 では $M = 1,500,000$ である．

$\sigma_{\min}$ から $\sigma_{\max}(= 1)$ まで選択率を変化させ，それぞれの選択率について PostgreSQL の問合せ最適化器の選択する問合せ実行計画を記録するとともに，当該計画のコスト見積値，および実行時間を測定した．

図 2 にその結果を示す．横軸は選択率，縦軸はコストおよび実行時間を秒単位で表したものである．各選択率において選択された問合せ実行計画について，選択条件の値以外は同一である問合せ実行計画は，同一パターンの問合せ実行計画であると見做し，各パターンを $P_i(i = 1, 2, \cdots)$ と表記する．図中で，薄い縦線が問合せ実行計画のパターンの変化点を表し，本実験では全部で 6 つのパターンが生成された．Q.3 においては，P_1 は索引走査およびネステッドループ結合が選択されており，P_5 以降は全表走査およびハッシュ結合が選択されている．P_2〜P_4 はそれらが組み合わさったパターンになっている．

図 2 に示す結果より，P_2 を除いた問合せ実行計画では，実行時間よりもコスト見積値の方が小さいことがわかる．また，P_1 の選択される領域において，選択率が小さいほど，コスト見積値が実行時間に比べ小さくなることがわかる．

図 3 は，Q-error の値をプロットしたものである．コスト及び実行時間のグラフから分かるとおり，P_1 の中でも選択率の小さい区間では，Q-error は大きくなっており，最大で 6.8 倍であった．一方，それ以外のプランでは 1.1〜1.6 倍であった．

図 4 は，コストと実行時間の差の絶対値をプロットしたものである．これにより，誤差が実際にどのくらい時間的影響を及ぼしているのかが分かる．グラフから分かるとおり，P_1 の中でも選択率の非常に小さな区間では，実行時間自体が小さいため，実際には時間的影響は小さいと言える．

以上により，索引走査およびネステッドループ結合を用いた問合せ実行計画は，Q-error は大きな値になるが，実行時間が小さいため，実際の時間的な影響は小さいと言える．また，全表走査およびハッシュ結合を用いたプランは，Q-error 自体は小さい値だが，実行時間が大きいため，実際の時間的な影響は大きいと言える．

3.3　TPC-H Q.1 – 10 を用いた評価

本実験では，TPC-H Q.1 から Q.10 を用いて，問合せ最適化器の選択する問合せ実行計画の各パターンについて E_{RMSE}，E_Q，$I \pm s$ を算出した．評価に際しては，元となる TPC-H の問合せに選択率調整を可能とする選択条件を追加し，3.2 節に示した実験と同様に，選択率を $\frac{100}{M}$%から 100%まで $M^{\frac{1}{N-1}}$ 倍

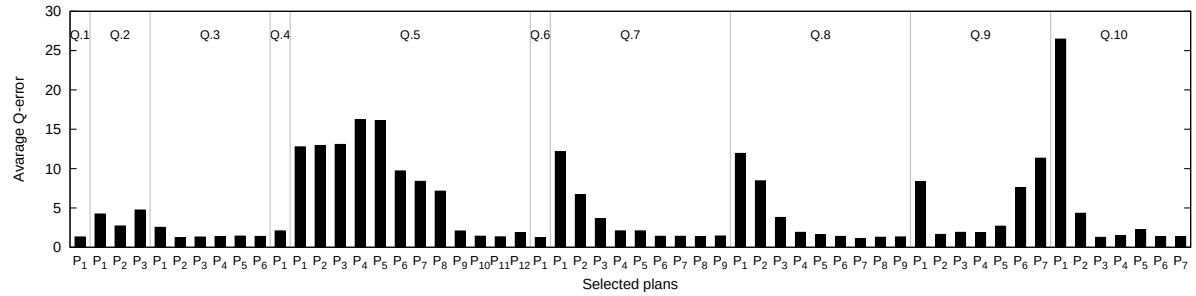

図 5　E_Q：問合せ・問合せ実行計画パターン毎の実行時間とコスト見積値の平均 Q-error
Fig. 5　Average Q-error between execution time and estimated cost for each query and plan pattern

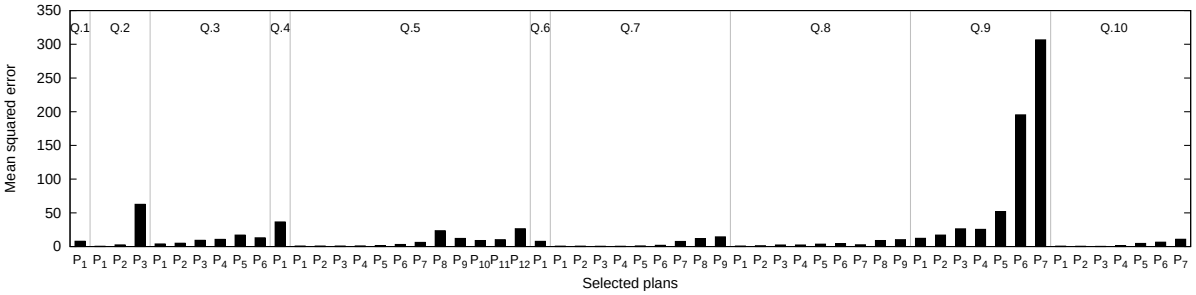

図 6　E_{RMSE}：問合せ・問合せ実行計画パターン毎のコストと実行時間の二乗平均平方根誤差
Fig. 6　Root mean squared error between execution time and estimated cost for each query and plan pattern

していき，測定を行った．M は選択率が 100%になるときに，問合せの条件式に指定する数値で，問合せごとに異なる．測定に用いた 10 個の問合せを，その特徴ごとに大きく 4 種類に分けることができる．

- 単一表に対する問合せ (Q.1, Q.6)
- 複数表の結合を含む問合せ (Q.3, Q.5, Q.7, Q.8, Q.9, Q.10)
- 副問い合わせを含む問合せ (Q.2)
- 準結合を含む問合せ (Q.4)

図 5，6，7 に結果を示す．図 5 は，各問合せ・各問合せ実行計画パターンについて E_Q を算出したものである．また各問合せににおいて，選択された最小の選択率が小さい問合せ実行計画パターンから順に $P_1, P_2, \cdots$ とした．同様に図 6 は E_{RMSE} を算出したものである．

2 つのグラフから，多くのクエリにおいて，選択率が低いほど E_Q は大きくなり，選択率が高くなると E_{RMSE} が大きくなる傾向が読み取れる．

単一表のクエリ（Q.1, Q.6）においては，いずれも全表走査の問合せ実行計画が選択されている．測定結果からは，E_Q，E_{RMSE} ともに小さく，誤差が小さいことがわかる．

複数表の結合を含む問合せ (Q.3, Q.5, Q.7, Q.8, Q.9, Q.10) では，選択率の小さな領域では索引走査およびネステッドループ結合を用いたプランが選択され，選択率の大きな部分では全表走査およびハッシュ結合を用いたプラン選択された．また，中間部分ではこれらの走査や結合方法が混在した問合せ実行計画が選択された．測定結果からは，索引走査およびネステッドループ結合の問合せ実行計画では E_Q が大きくなる傾向があるが，E_{RMSE} の値を見ると実行時間に与える影響は大きくないことが分かる．また，全表走査およびハッシュ結合の問合せ実行計画は，E_Q は小さいが実行時間自体が大きいため，実行時間に与える影響は大きく，特に Q.9 では顕著である．これは前節で考察した結果がそのまま当てはまるといえる．

副問い合わせを含む結合 (Q.2) では E_Q が 2.7%から 4.7%の間にあり，選択率の多い部分で E_{RMSE} が大きくなっている．この問合せでは，主問い合わせが生成する行数分繰り返し副問い合わせが実行されるが，キャッシュ効果がコストに十分反映されておらず，コストが大きく見積もられてしまっているのが一つの原因と考えられる．

準結合を含むもの (Q.4) は，他の問合せ実行計画と比べて比較的 E_{RMSE} が大きい結果となった．

3.4　考　　察

E_Q，E_{RMSE}，$I \pm s$ を算出することで，各問合せ実行計画の選択率ごとの誤差が定量化され，単一の指標ではとらえられない誤差が，他の指標と組み合わせることで明らかになる場合があることも分かった．

例えば，単一の表に対する索引走査の誤差はどの指標で見ても非常に小さいと言えるが，結合を含む問合せの選択率の低い領域では索引走査およびネステッドループ結合の E_Q は大きくなる傾向がある．つまり，より複雑な結合や副問合せにより大量に索引走査が行われる場合は大きな誤差につながる可能性があると予測できる．これは実行時間や E_{RMSE} のみではとらえられず，異なる指標である E_Q を併せて算出する意義を示して

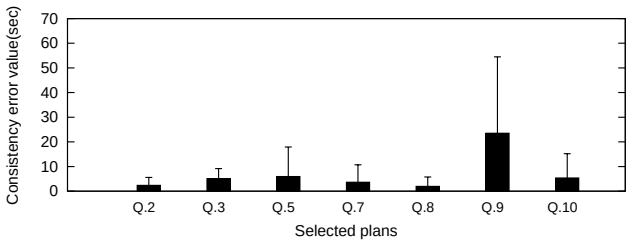

図 7　$I \pm s$：問合せ実行計画パターン損益分岐点における実行時間の非連続性

Fig. 7

いると言えよう．

このように，本論文で提案する複数の指標により，多角的にコスト見積もりの誤差を定量化する有効性が確認された．

4. 関 連 研 究

今日のデータベースシステムにおいてはコストベース最適化が広く普及しており，System R [2] に端を発するコストモデルによるコスト見積りと，当該コスト見積値に基づく問合せ実行計画の探索という伝統的な枠組みがその基本となっている．新たなハードウェア環境や問合せ実行方式の登場に伴い，新たなコストモデルも数多く提案が行われている [8]〜[11]．これらコストモデルの提案においては，当該モデルの精度に関する議論は行われているものの，問合せ最適化器が生成しうる相異なる問合せ実行計画間での整合性を含めた定量化はなされていない．

問合せ最適化器のプラン選択精度はデータベースシステムの実効的な性能に大きく影響するため，その精度向上や品質保証に関しては数多くの取り組みがなされている．[4] では，規定された性質を満たすデータベース及び問合せを生成することで，問合せ最適化器の精度を評価する手法が示されている．[12] では，カーディナリティ推定改善のベースラインとなる，正確なカーディナリティの値を，問合せ実行プランの全てのノードに対して効率的に算出する手法を提案している．[13] では，SQL Server におけるカーディナリティ推定アルゴリズムを全面的に刷新した際の品質保証のアプローチを示している．[3] では，問合せ問合せ最適化器によるプランのランク付けによって，複数の問合せ最適化器の精度比較を行う手法を提案している．また [14] は，結合順序探索のベンチマークを規定し，カーディナリティ推定の正確性が結合順序探索において極めて大きな影響を与えることを示した．一方で，コストモデルの誤差及びコストモデル間の整合性を定量化するという本論文のアプローチは，我々の知る限りにおいて行われていない．

5. 結　　論

本論文では，問い合わせ最適化器におけるコスト見積もりの正確性を評価する指標を提案し，実験的にその有効性を検討した．

コストモデルを評価する指標として，E_{RMSE}，E_Q，および $I \pm s$ の 3 つを提案し，実際に PostgreSQL の最適化器の評価を行ったところ，単一の指標のみではとらえられない誤差を相補的に検出可能であるという結果が得られ，その有効性を確認した．

文　　献

[1] S. Chaudhuri, “An overview of query optimization in relational systems,” Proceedings of the Seventeenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, pp.34–43, PODS ’98, ACM, New York, NY, USA, 1998.

[2] P.G. Selinger, M.M. Astrahan, D.D. Chamberlin, R.A. Lorie, and T.G. Price, “Access path selection in a relational database management system,” Proceedings of the 1979 ACM SIGMOD International Conference on Management of Data, pp.23–34, SIGMOD ’79, ACM, New York, NY, USA, 1979.

[3] F.M. Waas, L. Giakoumakis, and S. Zhang, “Plan space analysis: An early warning system to detect plan regressions in cost-based optimizers,” Proceedings of the Fourth International Workshop on Testing Database Systems, pp.2:1–2:6, DBTest ’11, ACM, New York, NY, USA, 2011.

[4] M. Stillger and J.C. Freytag, “Testing the quality of a query optimizer,” IEEE Data Eng. Bull., vol.18, no.3, pp.41–48, 1995.

[5] L.F. Mackert and G.M. Lohman, “Index scans using a finite lru buffer: A validated i/o model,” ACM Trans. Database Syst., vol.14, no.3, pp.401–424, Sept. 1989.

[6] G. Moerkotte, T. Neumann, and G. Steidl, “Preventing bad plans by bounding the impact of cardinality estimation errors,” Proc. VLDB Endow., vol.2, no.1, pp.982–993, Aug. 2009.

[7] H. Hacigumus, Y. Chi, W. Wu, S. Zhu, J. Tatemura, and J.F. Naughton, “Predicting query execution time: Are optimizer cost models really unusable?,” Proceedings of the 2013 IEEE International Conference on Data Engineering (ICDE 2013), pp.1081–1092, ICDE ’13, IEEE Computer Society, Washington, DC, USA, 2013.

[8] P. Ghodsnia, I.T. Bowman, and A. Nica, “Parallel i/o aware query optimization,” Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data, pp.349–360, SIGMOD ’14, ACM, New York, NY, USA, 2014.

[9] F.R. Reiss and T. Kanungo, “A characterization of the sensitivity of query optimization to storage access cost parameters,” Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data, pp.385–396, SIGMOD ’03, ACM, New York, NY, USA, 2003.

[10] H. Boral, W. Alexander, L. Clay, G. Copeland, S. Danforth, M. Franklin, B. Hart, M. Smith, and P. Valduriez, “Prototyping bubba, a highly parallel database system,” IEEE Transactions on Knowledge and Data Engineering, vol.2, no.1, pp.4–24, March 1990.

[11] W. Du, R. Krishnamurthy, and M.-C. Shan, “Query optimization in a heterogeneous dbms,” Proceedings of the 18th International Conference on Very Large Data Bases, pp.277–291, VLDB ’92, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1992.

[12] S. Chaudhuri, V. Narasayya, and R. Ramamurthy, “Exact cardinality query optimization for optimizer testing,” Proc. VLDB Endow., vol.2, no.1, pp.994–1005, Aug. 2009.

[13] C. Fraser, L. Giakoumakis, V. Hamine, and K.F. Moore-Smith, “Testing cardinality estimation models in sql server,” Proceedings of the Fifth International Workshop on Testing Database Systems, pp.12:1–12:7, DBTest ’12, ACM, New York, NY, USA, 2012.

[14] V. Leis, A. Gubichev, A. Mirchev, P. Boncz, A. Kemper, and T. Neumann, “How good are query optimizers, really?,” Proc. VLDB Endow., vol.9, no.3, pp.204–215, Nov. 2015.