5L-04

マルチバージョン索引構造 P-Tree の性能評価

野原 健汰 † 杉浦 健人 †† 石川 佳治 ††

† 名古屋大学情報学部コンピュータ科学科 †† 名古屋大学大学院情報学研究科

1 はじめに

現代のデータ駆動型アプリケーションでは，データベースには OLAP（Online Analytical Processing）と OLTP（Online Transaction Processing）の両方が求められる．データベース管理システムでの高速な読み書きの両立は，読み取りを改善するための最適化が書き込みのオーバーヘッドを引き起こすことが多く困難である．例えば，読み取りを改善するための方法として列指向ストアや実体化ビューがあるが，これらのアプローチは一般に書き込みを遅くする．列指向ストアは OLTP で行われる行単位での挿入や削除をより高価にし，実体化ビューは書き込み時にビューを無効化する必要があり書き込みのオーバーヘッドを大きくする．この問題の一つの解決方法として，マルチバージョン同時実行制御にスナップショット分離を使用するという方法がある．

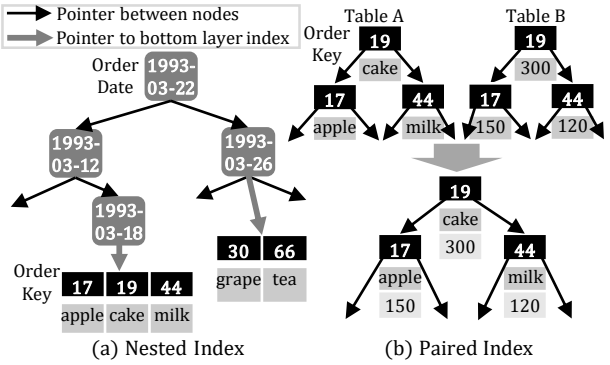
Parallel Binary Tree（P-Tree）[1] はこれを利用した手法であり，スナップショット分離とマルチバージョン同時実行制御をサポートする Hybrid Transaction/Analytical Processing（HTAP）データベース向けの索引として提案された．P-Tree は，パスコピーを使用して高速なマルチバージョン更新を行うイミュータブルなデータ構造である．また，効率的な書き込みを可能にしながら読み取りを向上させるために木の入れ子構造をサポートしている．このデータ構造は，索引とその基になるテーブルに対する一括操作のための並列アルゴリズムを可能にする．

P-Tree は，各種ベンチマークで最新の索引構造やデータベース管理システムを上回る結果を残したものの，索引としての基礎的な性能評価は十分であるとはいえない．そのため本稿では，P-Tree の再検証を行い，その性能特性などについて議論する．

2 P-Tree の概要

本章では P-Tree の手法概要を述べる．まず P-Tree の基本構造について述べた後，P-Tree の特徴であるスナップショット分離・マルチバージョン同時実行制御，及び Nested/Paired Index について述べる．

■P-Tree の構造 まず，P-Tree の基本的なデータ構造について述べる．P-Tree はソートされた key-value のペアを持つ平衡二分探索木である．key 及び value には任意の型を持つことができ，各ノードはそれぞれ key-value のペア，2 つの子ポインタ，サブツリーのサイズ，ガベージコレクション用の参照カウンタを持つ．また，P-Tree は書き込みを行う際にコピーをとるという Copy on Write（CoW）戦略を取っており，既存データに変更を加えないイミュータブルなデータ構造である．



(a) Nested Index (b) Paired Index

図 1 Nested Index 及び Paired Index の概観

■スナップショット分離及びマルチバージョン同時実行制御 読み取りと書き込みを高速に行う手法の一つとして，マルチバージョン同時実行制御にスナップショット分離を使用するという方法がある．これにより，同時実行の書き込みに関係なく，読み取りが実行できるようになる．P-Tree は書き込み時に索引のスナップショットを作成するためにルートのポインタを取得し，ルートの参照カウンタをインクリメントすることによって索引の現在のバージョンを取得する．また，パスコピーを使用して，書き込み時に木の新しいコピーを作成することでマルチバージョン管理を実現する．これにより，補助的なデータ構造を必要とせず，索引自体がバージョン履歴となる．

■Nested Index 及び Paired Index 効率的な読み取りを実現するために，P-Tree では Nested Index 及び Paired Index をサポートしている．Nested Index 及び Paired Index の概観は図 1 に示す．Nested Index は複数の索引を階層的に重ねたものであり，上層の索引のリーフノードから下層の索引へのポインタを持つ．例えば，主キーに対して索引付けされたテーブルに対し二次索引を追加したとき Nested Index は構築される．主キー側の索引は二次索引の値によって分割され，二次索引のリーフノードから参照される．一方，Paired Index は key を共有する二つのテーブルに対して一つの索引を使用するものである．これはテーブル結合の索引上での事前実行と考えられるが，ビューを実体化しないため，追加の空間や一貫性の問題を考えなくても良いという利点がある．

Performance Evaluation of P-Tree, a Multi-version Index Structure
Kenta Nohara†, Kento Sugiura††, and Yoshiharu Ishikawa††
†Department of Computer Science, School of Informatics, Nagoya University
††Graduate School of Informatics, Nagoya University

Copyright ©2021 Information Processing Society of Japan.
All Rights Reserved.

3 P-Tree の詳細

本章では，P-Tree の実装について述べる．

■直列化可能性のサポート スナップショット分離を実現するために，P-Tree 上においても複数トランザクションによる書き込みをサポートする必要がある．P-Tree ではパスコピーによる木構造の更新を行うため，単純な実装では各書き込みがそれ自身のコピーを作成する可能性があり，データの一貫性を保証できない．

この問題に対し，P-Tree ではグループコミットの一部として索引構造をバッチ更新する．いくつかのデータベースは複数のトランザクションをまとめてコミットするグループコミットをサポートしている．グループとしてコミットされるトランザクション群は何かしらの同時実行制御により直列化可能性が保証されており，書き込みセット内に競合は存在しない．P-Tree はそのような書き込みセットがバッチで与えられる状況を想定し，それらを並列で更新する方法を提供している．

■並列アルゴリズム P-Tree は，分割統治アルゴリズムを用いて書き込み及び読み取りの並列処理を実現している．並列タスクの粒度を制御するために，P-Tree は現在のツリーサイズがある閾値よりも大きい場合にのみ，2 つの再帰的な呼び出しを並列に実行する．それ以外の場合は逐次処理をする．これは，小さなタスクをフォークして結合する際のオーバーヘッドを避けるためである．この粒度は調整可能で，単一のツリーノードが扱うベースケースのワークロードに応じて決定される．

書き込み操作は，先述したバッチ更新をサポートする．まずバッチ内の key の重複を削除した後，key を基準にソートする．次にルートの key より小さい部分は左の部分木に，大きい部分は右の部分木にそれぞれ与え，再帰的に処理する．この 2 つの再帰処理は並列に動作する．

読み取り操作は，書き込み時と同様に並列で木を再帰的に探索し，そのスナップショットを返す．つまり，元の木構造そのものを返すのではなく，選択条件などを満たす部分木のコピーを返す．例えば範囲検索であれば，範囲の境界となる部分のみパスコピーによる索引構造の修正を行い，範囲内のデータのみを指す索引構造のルートポインタを返す．この読み取り方法は索引構造の更新に必要な部分のスキャンのみで済み，対象データ全体のスキャンを避けられる．更に，後続の演算は索引を利用してデータにアクセスできるため，データそのものを返すよりも効率的な処理が期待できる．

■ガベージコレクション P-Tree は，各ノードがもつ参照カウンタをもとにガベージコレクションを行う．参照カウンタは各ノードが参照されている数を保持している．DBMS がスナップショットを削除したときにルートの参照カウントをデクリメントする．ルートの参照カウンタが 0 になったとき，ルートの領域を解放する．その後，ルートの 2 つの子を並列に再帰的に取得する．ルートを解放すると参照カウントが 0 になるため，参照カウントが 1 のツリーノードは領域を解放する．これは，収集されたスナップショット上にのみある全てのツリーノードを対象とする．つまり，他のスナップショットの他のツリーノードには影響しない．

4 評価分析

本章では，元論文 [1] で行われた P-Tree の実験概要の説明とその不足点を補う基礎性能の評価方法を説明する．

■元論文で行われた実験概要 元論文では，OLTP 性能評価，OLAP 性能評価，HTAP 性能評価の 3 つの実験が行われた．

OLTP 性能評価では，YCSB（Yahoo! Cloud Serving Benchmark）[2] の Insert-only，Read/Update（95/5），Read/Update（50/50），Read-only の 4 つのワークロードを用いて実験が行われた．5 回の実行時間の平均と標準偏差を記録した．

OLAP 性能評価では，TPC-H ベンチマークを用いて実験が行われた．主索引及び二次索引を保持するために，7 つのネスト構造をもった P-Tree を使用した．各問合せについて，5 回の幾何平均を記録した．

HTAP 性能評価では，TPC-H と TPC-C の 3 つのトランザクションを統合した TPC-HC ベンチマークを用いて実験が行われた．1 スレッドは書き込み用，その他のスレッドは読み取り用として使用した．同時書き込みを行った上で，各問合せについて 5 回の幾何平均を記録した．また，3 つのトランザクションのスループットを計測した．

■基礎性能の評価方法 元論文で行われた実験では，基礎的な性能評価が不足している．そのため P-Tree の基礎性能評価として，主索引や二次索引を単体で構築した場合の簡単な問合せに対する挙動を確認する．具体的には，読み取り時の点検索及び範囲検索の実行時間やその際の各ノードの参照回数，書き込み時の実行時間やメモリ使用量などを計測する．また，YCSB や TPC-H のような既存のベンチマークでも評価を行う．

5 まとめと今後の課題

本稿では，スナップショット分離とマルチバージョン同時実行制御をサポートする HTAP データベース向けの索引構造である P-Tree の手法概要及び実装詳細について述べた．今後は，4 章で述べたように P-Tree の基礎性能評価のための実験を行っていく予定である．

謝辞

本研究は JSPS 科研費（16H01722, 19K21530，20K19804）の助成，及び国立研究開発法人新エネルギー・産業技術総合開発機構（NEDO）の委託業務（JPNP16007）から得られた結果による．

参考文献

[1] Y. Sun, G. E. Blelloch, W. S. Lim, and A. Pavlo, “On supporting efficient snapshot isolation for hybrid workloads with multi-versioned indexes,” *PVLDB*, vol. 13, no. 2, pp. 211–225, 2019.

[2] B. F. Cooper, A. Silberstein, E. Tam, R. Ramakrishnan, and R. Sears, “Benchmarking cloud serving systems with YCSB,” in *Proc. SoCC*, pp. 143–154, ACM, 2010.

Copyright ©2021 Information Processing Society of Japan.
All Rights Reserved.