2N-3

分散インデックス手法の負荷分散処理比較のための共通フレームワークの構築

徳田 聡介† 渡辺 陽介†† 横田 治夫‡

† 東京工業大学 工学部 情報工学科 †† 東京工業大学 学術国際情報センター
‡ 東京工業大学 大学院情報理工学研究科 計算工学専攻

1 はじめに

大量のデータを複数の計算機で管理するために分散インデックス手法が数多く提案されている [1][2][3]。各手法を一から実装することは開発者にとって労力を要することであるため、これまで各分散インデックス手法は共通の環境での比較は十分にされてこなかった。そこで我々は分散インデックス手法を比較するための共通フレームワークを構築している [4]。分散環境では、特定の計算機がボトルネックとならないようにする負荷分散が重要であり、分散インデックス手法での対応が必要となる。これまでフレームワーク上では完全一致、範囲問い合わせなどの共通化を実現してきたが、負荷分散処理についての共通化はなされてこなかった。本研究では、負荷分散処理のうち各手法のデータ構造に非依存の処理をフレームワーク側で提供し、開発者の実装作業を容易とすることを目的とする。

2 共通フレームワーク

提案フレームワークにおいては、開発者は、各手法固有のデータ構造に基づく処理のみを既定のメソッドとして実装するだけでよく、共通処理についてはフレームワーク実行システム側で事前に提供している [4]。以降、フレームワーク実行システムのことを単に実行システムと呼ぶ。

2.1 対象とする分散インデックス手法

本稿では以下の手法を共通化対象とする。

SkipGraph[1] Skip lists をベースにしている。複数の層に分かれたリスト構造をしており、上位の層からリンクを辿り目的の計算機を探す (図 2, 図 3)。

P-Ring[2] 分散ハッシュテーブル上に Hierarchical Ring(HR) とよばれる表を構築しており、この表で他の計算機の担当範囲を管理する (図 4, 図 5)。

FatBtree[3] キー空間を各計算機に均等に配分し、葉ノードからみて上位にあたるノードのみを計算機に配置する。根から葉ノードへの探索時にノードが途切れていればそのノードを持つ計算機にアクセスする。

2.2 フレームワーク実行システムの構成

分散環境内の各計算機で動作する実行システムの構成を図 1 に示す。受付モジュールがメッセージを受け取り、完全一致、範囲問い合わせなどの処理スレッドを作る。メッセージ通信モジュールはメッセージの送受信を行う。新規追加した負荷分散モジュールで負荷分散の手法共通処理を提供する。手法依存の処理はインターフェースモジュールを通じて SkipGraph、P-Ring、FatBtree などとやりとりする。

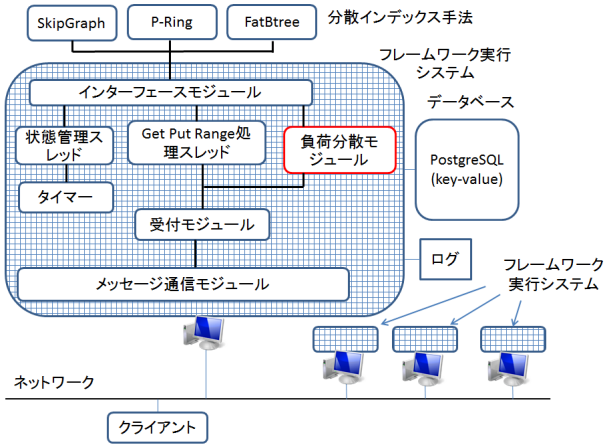

図 1: フレームワーク実行システム

3 負荷分散処理の共通化部分

実行システム側で共通化された処理は、負荷の大きい計算機の特定、データ転送の対象となる計算機の決定、転送するデータの決定、各手法の selectElementToLock メソッドを呼び出して得た要素への書き込みロック、メッセージ通信、ロック情報の保持、ロック解放である。実行システムは、各手法のデータ構造に依存した処理を行うために必要なメソッドを提供しており (表 1)、これを呼び出すことで手法固有の処理を行う。

以下、中心となるデータ転送とインデックス更新について述べる。

3.1 データ移動

まず、実行システムが各手法に selectElementToLock メソッドを呼び出し、ロックする要素を取得し書き込みロックをかける。次に、実行システムは makeLockStrategy メソッドを呼び出してロック情報を取得し、

A Framework for comparing load balance processes of distributed index methods

Sosuke Tokuda†, Yousuke WATANABE††and Haruo YOKOTA‡
†Department of Computer Science, School of Engineering, Tokyo Institute of Technology
††Global Scientific Information and Computing Center, Tokyo Institute of Technology
‡Department of Computer Science, Graduate School of Information Science and Engineering
{tokuda, watanabe}@de.cs.titech.ac.jp, yokota@cs.titech.ac.jp

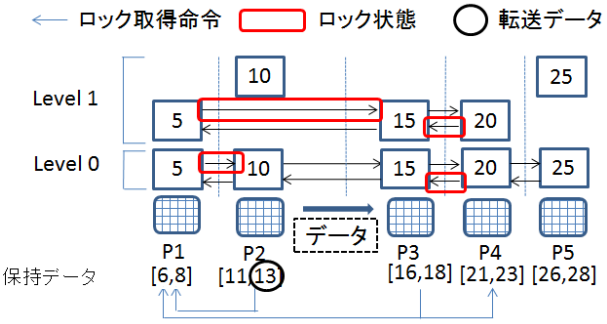

図 2: SkipGraph データ移動前 (P2 の 13 を P3 へ)

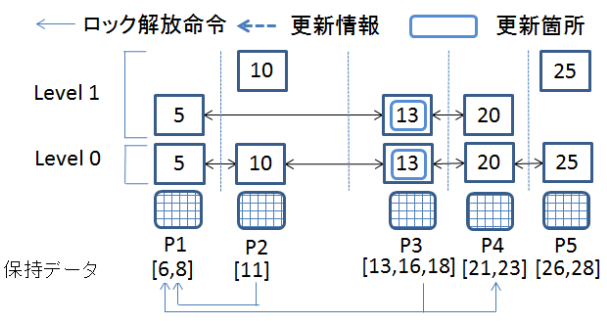

図 3: SkipGraph データ移動後 (P2 の 13 を P3 へ)

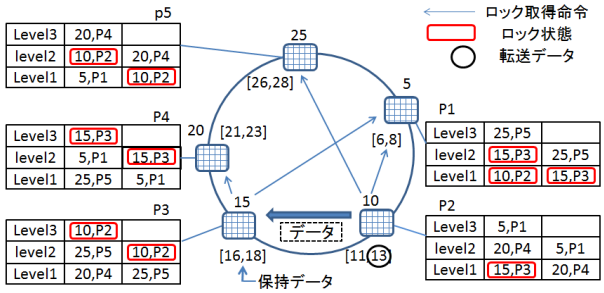

図 4: P-Ring データ移動前 (P2 の 13 を P3 へ)

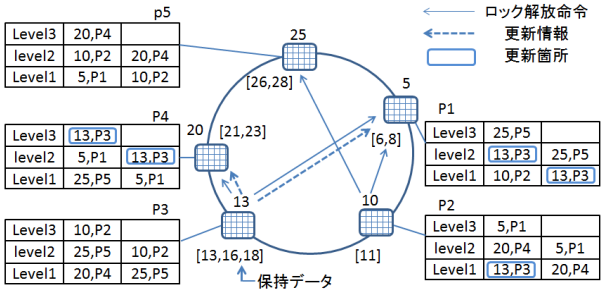

図 5: P-Ring データ移動後 (P2 の 13 を P3 へ)

送り先に指定された計算機へとロック情報を転送する。そして、実行システムが各手法へ makeMoveStrategy メソッドを呼び出し、データ情報を受け取った後、データ送り手と受け手の間でメッセージ通信する。

図 2、図 4 で, SkipGraph、P-Ring 上で P2 から P3 へキー 13 を移動する前の様子を示した。SkipGraph は P2、P3 で makeLockStrategy メソッドが呼び出された時に、リンクしている P1、P3 へのロック情報を作成する。それに対し、P1、P3 は P2、P3 へのリンクをロックする。P-Ring の場合は、P2、P3 は自身へのポインタを持つ P1,P5 と P1、P4 へロック情報を送る P1、P5、P4 は P2 または P3 へのポインタをロックする。

3.2 インデックス更新

データを移動すると計算機の担当するデータ範囲が変わるので、更新された担当範囲情報を周囲へ伝搬する。実行システムは updateIndex メソッドを呼び出し、各手法から更新情報を取得する。そして送り先に指定された計算機へと更新情報を転送する。

図 3、図 5 で、SkipGraph、P-Ring 上で P2 から P3 へキー 13 を移動した後の様子を示した。SkipGraph は他の計算機の担当範囲の情報を保持していないため、P2、P3 から他の計算機へ更新情報の転送はない。P-Ring の場合は、P3 の担当範囲が変化するため、P3 が P3 へのポインタをもつ P1、P4 へ更新情報を転送する。

4 まとめと今後の課題

本論文では、我々が提案しているフレームワーク上に負荷分散処理を共通化するモジュールを追加し、各手法のデータ構造に非依存の部分を共通化した。それにより、手法ごとの負荷分散処理を公正に比較できるようになり、また各手法を比較するための実装を容易

表 1: 分散インデックス実装に必要な主なメソッド

名前	役割
selectElementToLock	自身からロックが必要な要素の選択
makeMoveStrategy	データ受け手へ転送するデータ情報を作成
makeLockStrategy	データ転送に関与しない計算機へ転送するロック情報を作成
updateIndex	インデックスの更新

にした。今後の課題として、各手法の負荷分散処理における性能を比較することがあげられる。

参考文献

[1] J.Aspnes and G.Shah. Skip graphs. *ACM Transactions on Algorithms*, Vol. 3, No. 4, p. 37, November 2007.

[2] A.Machanavajjhala J.Gehrke J.Shanmugasundaram A.Crainiceanu, P.Linga. P-ring: An efficient and robust p2p range index structure. In *SIGMOD*, 2007.

[3] H.Yokota, Y.Kanemasa, and J.Miyazaki. Fat-btree: An update-conscious parallel directory structure. In *Proceedings of the 15th International Conference on Data Engineering, Sydney, Australia, March 23-26, 1999*, pp. 448–457. IEEE Computer Society, 1999.

[4] 近藤直樹, 羅敏, 渡辺陽介, 横田治夫. 範囲問合せ可能な分散インデックスの性能評価. *DEIM2011*, 2011.