様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９　（共通）

科学研究費助成事業　　研究成果報告書

平成　３０　年　　４　月　２１　日現在

機関番号：　１３９０３
研究種目：　基盤研究(C)（一般）
研究期間：　2015～2017
課題番号：　１５Ｋ００１６８

研究課題名（和文）複数一貫性レベルに対応したスケールアウト可能なデータベースリプリケーション

研究課題名（英文）Acceleration of Back-end Storage by Relaxing Consistency

研究代表者
松尾　啓志（MATSUO, HIROSHI）

名古屋工業大学・工学（系）研究科（研究院）・教授

研究者番号：００２１９３９６

交付決定額（研究期間全体）：（直接経費）　　3,500,000円

研究成果の概要（和文）：データーベースリプリケーションプロトコルにおいて、複数の一貫性をサポートすることはスループットのスケールアウトの面から重要である。従来手法であるMcRepは、レプリカの前段にプロキシーを設置することにより、複数の一貫性を提供した。しかしこの手法は、プロキシーの性能によりスケールアウトしないことは明らかである。本研究では、レプリカの後段にレプリケータを配置することにより、スケールアウト可能なレプリケーションプロトコルの提案を行った。

研究成果の概要（英文）：In the database replication protocol, supporting multiple consistencies is important from the viewpoint of scale-out of throughput. McRep, the conventional method, provided multiple consistency by placing a proxy in front of the replica. However, it is obvious that this method does not scale out due to the performance of the proxy. In this research, we proposed a replication protocol that can be scaled out by placing a replicator at the end of the replica.

研究分野：　分散システム

キーワード：　データーベース　リプリケーション　一貫性制御

１．研究開始当初の背景

FacebookやAmazonに代表されるように、今や１０億人が同時に１つのデータベースにアクセスする時代が到来した。このような場合、NoSQL と呼ばれる分散キーバリューストア(以下 D-KVS)型のデータベースの利用が一般的である。しかし、複雑な処理（特に更新時）を実現する際には、トランザクション処理や関係演算を行うことのできるリレーショナル DB(RDB)が必要となる。従来から RDB を高性能化する手法として、リプリケーションが用いられている。しかし特に更新時の処理能力とスケーラビリティに問題があった。Raihan Al-Ekram らが提案した※複数一貫性データリプリケーション方式(McRep 方式)は、この問題を解決することが期待できる。しかしクライアントからの呼び出し処理を受け付けるプロキシノードが必要なため、性能がスケールアウトできないことは明らかである。

従来我々は、D-KVS で各ノードを自律的に協調動作させるスケジューリング手法を開発している。その過程において、McRep 方式の弱点であるプロキシノードを排除し、各ノードが自律分散的にリプリケーションを行う新しい手法を着想するに至った。

２．研究の目的

ビックデータの時代を迎え、データベースシステムの高速化は必須である。分散キーバリューストア型データベースはその特性上スケールアウトが容易である。しかし、複雑な検索が可能であるリレーショナルデータベースシステムを複数ノードにより構成することによる高速化（スケールアウト）は、非常に困難であることが知られている。

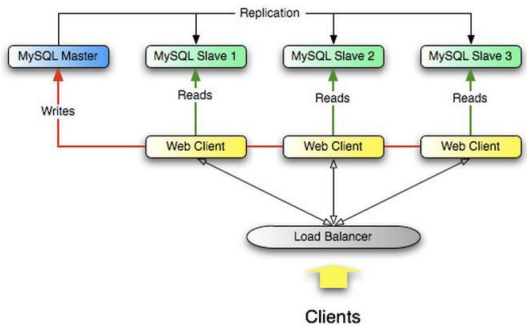


図１ MySQL に於けるデータリプリケーション　読み込み時はリプリケーションデータにアクセス可能なので、性能はスケールアウトする。しかし更新処理は、Master(左)ノードに対してしか行えないので、スケールアウトしない。

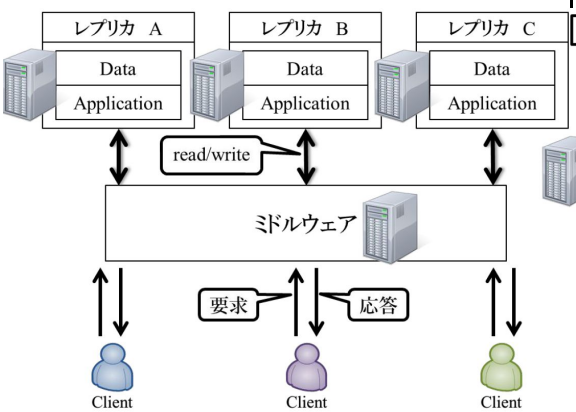


図２　McRep 方式　クライアントからの要求はミドルウエアを実行するプロキシノードを中継する必要あり

図１は代表的な RDB である MySQL のデータリプリケーション時の検索クエリーの流れである。読み出しには、MySQL Slave1～3 に検索クエリーを送信可能であるが、更新時は一台の MySQL Master のみが行うので、スケールアウトは不可能である。

一方、McRep 方式は、データベースシステムの前段にミドルウエア層を挿入するとともに、各ノード内のアプリケーションが、ミドルウエアと協調しリプリケーションプロセスを制御することにより、複数のレプリカに並列に書き込むことが可能なプロトコルを開発した（図２）。提案されたプロトコルは、様々な一貫性（直列化可能性, 順序,セッションスナップショット,１コピー直列化, 因果）にも対応可能である。さらに計算機シミュレーションにより、更新時を含む性能が向上することを確認した。

しかし、図２に示す関係から明らかなように、ミドルウエアを実行するノードにデータベース参照・更新要求が集中する。従って、データベースノード台数を増加させても、ミドルウエアを実行するノード（プロキシーノード）の性能に変化はないため、クライアントからの要求が大量となった場合、処理することができなくなる。つまり性能のスケールアウト性に問題がある。

一方、図３に示す提案方式では、レプリカが存在するノードに対して直接クライアントからのデータベース参照・更新要求が行われる。つまりプロキシノードを介する必要がないため、データベースノードの台数の増加に対して、性能がスケールアウトすることが期待される。しかし、さまざまな一貫性レベルでの動作を保証したリプリケーションを行うためには、多数のノード間での複雑かつ動的な環境下での分散制約充足・最適化問題を解く必要がある。

３．研究の方法

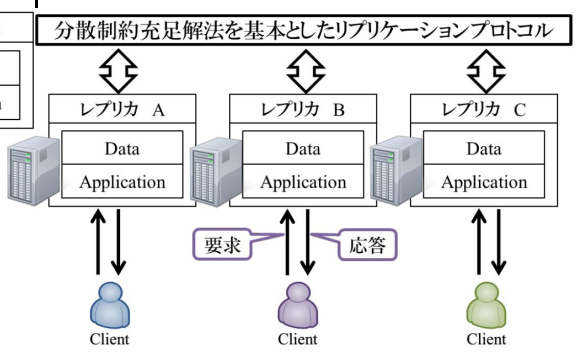


図３　提案方式　クライアントからの要求はミドルウエアを実行するプロキシノードを中継する必要がなく、レプリカノードに直接アクセス可能。

本研究は、以下の３つのプロジェクトに分割

して遂行した。

プロジェクト１：分散制約充足プロトコルによる協調リプリケーションシステム開発

従来から、多数のパラメータや動的な負荷環境が存在するクラスタシステムに於けるスケジューリング問題は、経験的な知見により行われる場合が多い。このプロジェクトでは、リプリケーション負荷などが変動する数十台のノードと、大量の情報検索パケットが存在する環境での協調リプリケーションを想定している。しかも従来からの経験的なアプローチではなく、理論的な枠組みに裏打ちされた手法の開発を目的とする。本プロジェクトを遂行するには、(1)リプリケーションアルゴリズムの開発・最適化、(2)プロトコルレベルの最適化の２つの課題を解決した。

プロジェクト２：PostgreSQL を用いた実装グループ

このプロジェクトでは、オープンソースとして公開されている PostgreSQL データベースと pgpool-II レプリケーションシステムを基本として、本研究で提案した協調リプリケーションアルゴリズムを実装した。Pgpool-II はプロキシ型のリプリケーション方式を採用しているが、我々はすでに、PostgreSQL と Pgpool を改造して、ノード間でのリプリケーションを行うシステムの開発を行った。

４．研究成果

4.1 複数の一貫性レベルを保証可能なバックエンドデータリプリケーション

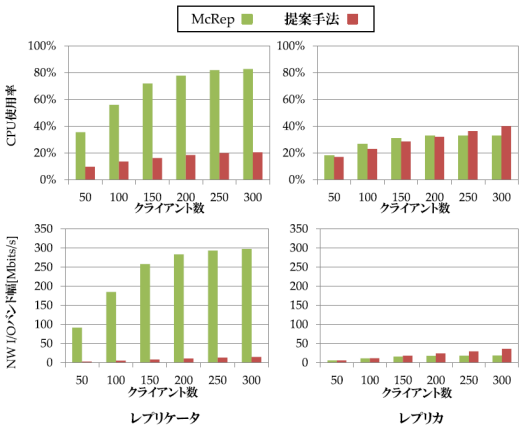
既存手法である McRep は、クライアントの全リクエストがリプリケータを経由するため、リプリケータの性能がボトルネックとなる問題点が存在した。提案手法は図３に示すように、レプリケータがミドルウエア部ではなく、バックエンド部で動作するように拡張する。提案手法では、さらにレプリケータだけでなく、レプリカにおいてもバージョン管理を行い、一貫性制御を行うように拡張を行った。これによりレプリケータが集中的に行っていた一貫性制御を分担して行うことにより、レプリケータの処理量が軽減される。さらにレプリケータがバックエンドで動作することで、設定されている一貫性レベルを満たすレプリカが存在する場合は、レプリケータが Read-only リクエストを処理する必要がないため、さらにリプリケータの処理量を削減できる。

評価環境を下表に示す。

	PostgreSQL	pgpool-II
version	9.3.5	3.3.3
OS	Linux 3.5.0-23-amd64 Ubuntu 12.04 Server	
CPU	Intel(R) Core(TM) i5 3470 @ 3.2 GHz	
Memory	16 GB	
Network	1000BASE-T	

4.2 評価結果

レプリカ数を固定してクライアント数を変化させてスループットを評価した。レプリカ数を６、pgbench のクライアント数を 50 から 300 まで変化させた場合、One-Copy



Serializability において、既存手法はクライアント数が 150 付近でスループットが頭打ちするのに対して、提案手法は 300 を超えてもスループットが向上することを確認した（Update トランザクションと ReadOnly トランザクションの比が 1:9 の場合）。

リプリケータとレプリカノードの CPU 利用率においても、下図に示す通り、提案手法適用時のレプリケータの CPU 利用率を激減させることを確認した。

５．主な発表論文等

（研究代表者、研究分担者及び連携研究者には下線）

〔雑誌論文〕（計　2 件）

①Atsushi Ohta, Ryota Kawashima and Hiroshi Matsuo", A Replication Protocol Supporting Multiple Consistency Models without Single Point of Failure, IEICE Transactions on Information and Systems, 査読あり, vol.E99-D, no.12, pp.3013-3023,(2017)
DOI: 10.1587/transinf.2016PAP0014

②太田 篤, 松野 雅也, 川島 龍太, 松尾 啓志," 複数の一貫性レベルを保証可能なバックエンドベースデータレプリケーション",査読あり, 情報処理学会論文誌, vol.57, no.3, pp.812-822,(2016)
https://ipsj.ixsq.nii.ac.jp/ej/?action=repository_uri&item_id=158113&file_id=1&file_no=1

〔学会発表〕（計　4 件）

①稲垣 英夫, 川島 龍太, 松尾 啓志," Apache Spark の Serialize 処理最適化による処理速度向上手法",情報処理学会オペレーティングとシステムソフトウエア研究会,Vol.2017-OS-141,No.16,pp.1-7(2017)

②川浪 大知, 川島 龍太, 松尾 啓志," 分散 KVS におけるアクセス頻度を考慮したコンパクション頻度の動的変更", 情報処理学会

オペレーティングとシステムソフトウエア研究会,Vol.2017-OS-141,No.19,pp.1-7(2017)
③鴨下 将成, 川島 龍太, 松尾 啓志," 分散キーバリューストアにおけるアクセス頻度を考慮した階層化ストレージ手法の提案", 情報処理学会オペレーティングとシステムソフトウエア研究会,Vol.2016-OS-138,No.16,pp.1-7(2016)
④太田 篤, 川島 龍太, 松尾 啓志," 複数の一貫性レベルを保証可能なマルチマスタデータレプリケーション", 平成 27 年度電気・電子・情報関係学会東海支部連合大会講演予稿集(2015)

６．研究組織

(1)研究代表者
松尾啓志　(HIROSHI MATSUO)
名古屋工業大学・大学院工学研究科・教授
研究者番号：00219396

(3)連携研究者
川島龍太　(KAWASHIMA RYOUTA)
名古屋工業大学・大学院工学研究科・助教
研究者番号：00710328