ハイブリッドクラウドにおける データベース同期に関する考察と提案

細谷 柚子[1] 三島 健[2] 小口 正人[1]

概要：近年，クラウドコンピューティングモデルの出現に伴いパブリッククラウドやプライベートクラウドが普及しつつあり，その両者をシームレスに結合するハイブリッドクラウドが注目されつつある．しかし，実社会においてはデータの一貫性を保たなければならず，そういった技術的な問題によりハイブリッドクラウドの導入はあまり進んでいない．他方で，データベースサーバは企業の基幹を構成しているため，クラウドで動作させるべき重要度の高いシステムである．そこで，本研究では，ハイブリッドクラウド環境でデータベースを同期させることに注目した．
LAN 環境を前提としてデータベースサーバを同期する Pangea という既存のミドルウェアがある．これをハイブリッドクラウドに適用し，TPC-W ベンチマークを用いて評価実験を行った．データセンタが遠隔地にあることを想定して，Dummynet を使って人工的に遅延を挿入した．近隣の街にバックアップを置く場合と，海外のような遠隔にバックアップを置く場合を想定し，遅延は RTT16ms と RTT256ms で測定した．LAN 環境における結果と 2 つの遅延時間における結果とを比較し，考察を行った．またその結果から性能向上を目指し，Pangea の修正・拡張を提案する．

A Consideration and Proposal on Database Replication in the Hybrid Cloud

YUZUKO HOSOYA[1] TAKESHI MISHIMA[2] MASATO OGUCHI[1]

1. はじめに

近年，クラウドコンピューティングモデルの出現に伴い，パブリッククラウドやプライベートクラウドが普及しつつある．さらに，それらのクラウドをシームレスに結合する形態のハイブリッドクラウドの検討も行われてきている．両者を併用するハイブリッドクラウドはそれぞれのリソースを必要に応じて使い分けることができ，現在注目されているシステムである．しかし，実社会においては，ハイブリッドクラウドの導入はあまり進んでいない．その理由の１つは，２つのクラウドを併用するためには，ハイブリッドクラウドにおけるデータの一貫性を保つことが重要であるが，それは技術的に難しい問題であるためである．

本研究ではハイブリッドクラウドにおいてデータベースを同期させることに注目した．データベースサーバは企業の基幹を成していることを考えると，クラウドの重要なテーマである．

LAN 環境を前提としてデータベースサーバを同期する Pangea[1] という既存のミドルウェアがある．これをハイブリッドクラウド環境に適用し，TPC-W ベンチマーク [2] を用いて評価実験を行った．実験では東京―大阪間を想定した RTT16ms と日米間や日欧間を想定した RTT256ms の場合を測定した．高遅延の測定は，外資系の企業や，日本企業の海外進出による海外支店が増えたことで，日本と海外でのデータの同期が必要になってくると考えたためである．また，日本のように自然災害が多い国では，近隣にバックアップを置いておくと，大規模災害が起こった際には両方とも失ってしまう恐れもあるため，遠隔バックアップの需要が高くなっていることも考えられる．以上の評価をもとに，ハイブリッドクラウドにおけるデータベース同期方式の課題の抽出とその解決法について検討を行う．

[1] お茶の水女子大学
Ochanomizu University
[2] NTT ソフトウェアイノベーションセンタ
NTT Software Innovation Center

2. ハイブリッドクラウド

クラウドコンピューティングにより提供されるサービスのうち，不特定多数を対象にしたオープンなクラウドをパブリッククラウド，自社内で構築されたクラウドをプライベートクラウドとする．パブリッククラウドではサーバリソースやアプリケーションサービスを必要な時に必要なだけ利用できる．ビジネスに合わせたスケールアウト/スケールダウンが可能であり，無駄なくリソースを使用でき，設計や運用のためのコスト削減が期待できる．またデータ管理を専門の業者に預けることにより，ユーザは技術面を気にすることなく利用でき，技術面のリスク削減にも繋がる．しかし，社外のサービスを利用することによるセキュリティへの不安の声も多くあり，これはクラウド導入が積極的に行われていないことの要因でもある．

他方でプライベートクラウドは，社内のシステムであるため，特定のネットワークや場所からのアクセスのみ許可されていることから，パブリッククラウドに比べるとセキュリティの不安は少ない．

この２つのクラウドを併用するハイブリッドクラウドでは，それぞれの持つ拡張性や安全性などのメリットを利用することができる．例えば，自社でデータセンタを保有し，通常はデータ管理に自社システムのプライベートクラウドを使用して，時期により短期間に大容量データ処理が必要になった場合やアクセスが急増したときの対応手段として，拡張性が高く迅速なパブリッククラウドを利用することが挙げられる．また，通常パブリッククラウドを使用している場合でも，セキュリティ面を考慮し，個人情報や社外秘の情報はプライベートクラウドで管理するといった利用方法も検討されており，現在までででハイブリッドクラウドの有用性は強調されている．

ハイブリッドクラウドでデータを管理するには，2 つのクラウドでデータベースの同期が行われていない場合，まず最新のデータベースにアクセスするという手間がかかり，処理性能が低くなってしまうことが懸念される．ハイブリッドクラウド上のデータベースの一貫性が保たれていることにより，どのデータベースにアクセスしても同一の結果を得ることができ，処理をスムーズに行えることが期待できる．そのため，データベースの同期はハイブリッドクラウドを使うための重要な課題であると考える．

3. Pangea

Pangea は図 1 に示すように，LAN 環境を前提として複数のデータベースサーバ間でスナップショット分離を保証するデータベースレプリケーションミドルウェアである．クライアントからサーバ側に直接アクセスするのではなく，ミドルウェアを介してデータベースにアクセスする方式となっていることから，データベースを改造することなく，サーバを増やすことで性能を向上させることが可能である．Pangea では照会処理は 1 台のレプリカで，更新処理は全レプリカで実行される．クライアントからミドルウェアまでの処理を Global transaction，ミドルウェアからサーバまでの処理を Local transaction といい，全レプリカでの Local transaction がコミットされ次第，Global transaction のコミット処理が完了する．また，レプリカの中の 1 台を Leader，その他は Follower とし，更新処理の場合は Leader に対して更新をした後に，他の Follower に対しても同様に処理を行う．

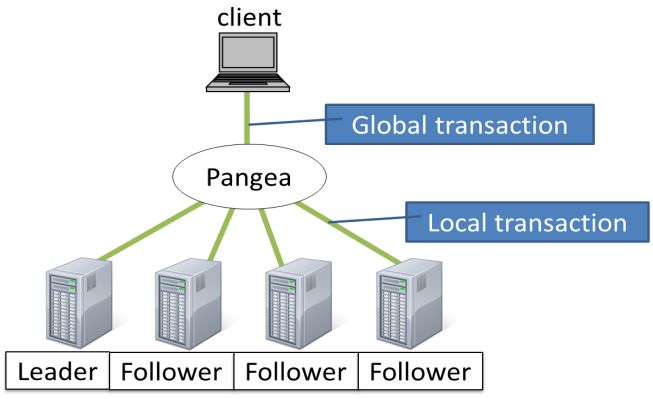

図 1 Pangea

本研究では Pangea の基礎性能測定をした後に，ハイブリッドクラウド環境での性能測定を行い，遅延による影響を分析した．また，その結果から Pangea の拡張を検討する．

4. 基礎性能評価

4.1 実験環境

Pangea の基本性能評価を行った．データベースサーバ用にマシン 2 台を用いた．どちらのマシンも表 1 に示すスペックである．データベースサーバには PostgreSQL9.2.6 を使用した．クライアントとデータベースサーバの間に Pangea を接続して同期を行う．Web サーバとアプリケーションサーバには Tomcat6.0.37 を用いた．性能評価は TPC-W ベンチマークを使用した．

TPC-W は Web ベースでトランザクション処理を行う，電子商取引を模擬したベンチマークで仮想的なブラウザ (以下 EB とする) が，データベースにトランザクションを発行する．データベースは item，country，author，customer，orders，order_line，cc_acts，address の 8 種のテーブルから成り，14 種のインタラクションがある．

TPC-W には 3 種類のワークロードがあり，それらの違いは表 2 に示すように照会処理と更新処理の割合が異な

る．性能評価指標は，スループット (1 秒あたりの Web 画面表示:(WIPS)) とレスポンス時間 (1 画面データの転送時間:(秒)) とした．TPC-W は平均 7 秒の Thinking Time が存在するため，次のリクエストを送るまでに，平均 7 秒の時間がかかる．そのため，リクエスト率は 1/7 リクエスト/秒となる．

また，パブリッククラウドは遠隔地にあることを想定して Dummynet を使用して人工的に遅延を発生させた．今回は東京－大阪間を想定した比較的低遅延の 16ms の場合と，日米間や日欧間を想定した高遅延の 256ms の場合で実験を行った．実験環境を図 2 に示す．

表 1　マシンのスペック

OS	Linux 2.6.32
CPU	Intel(R) Xeon(R) CPU @ 1.60GHz
Memory	2GByte

表 2　ワークロード

ワークロード	Read-only	Update
Browsing mix	95 %	5 %
Shopping mix	80 %	20 %
Ordering mix	50 %	50 %

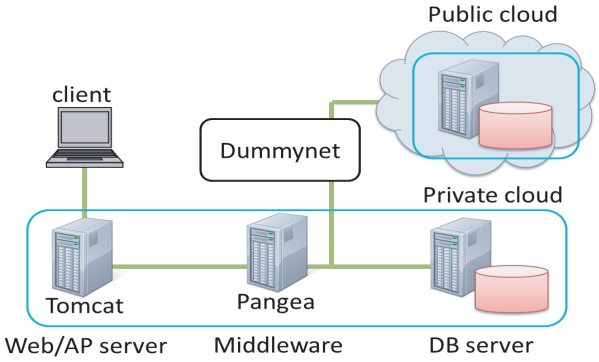

図 2　実験環境

4.2　基本性能

TPC-W の 3 種類のワークロードにおけるスループット (WIPS) とレスポンス時間 (秒) をそれぞれ図 3，4，5 に表す．上の折れ線グラフはスループット，下の折れ線グラフはレスポンス時間を表している．

最大スループットの値，そのときの EB 数ともに，ワークロードによって異なる．browsing mix では EB が 80 のとき最大スループットが 9.1WIPS，shopping mix では EB が 110 のとき最大スループットが 14.7WIPS，ordering mix では EB が 290 のとき最大スループットが 38.9WIPS となった．3 種類とも EB 数を増やしていくにつれ，スループットも上がっていく．特にレスポンス時間は指数関数的に上昇する．レスポンス時間が上昇していき，1 より大きくなるあたりでオーバーロード状態になるため，EB 数を増やしてもスループットは下降していく．

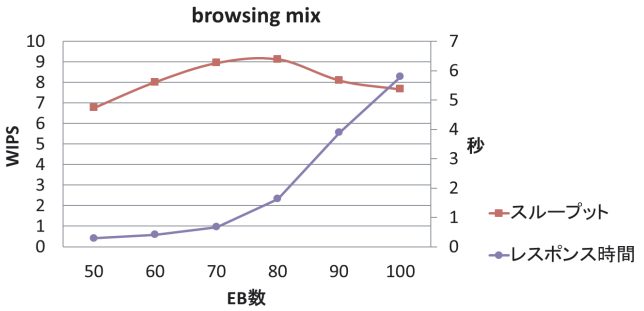

図 3　ローカル環境における browsing mix の性能

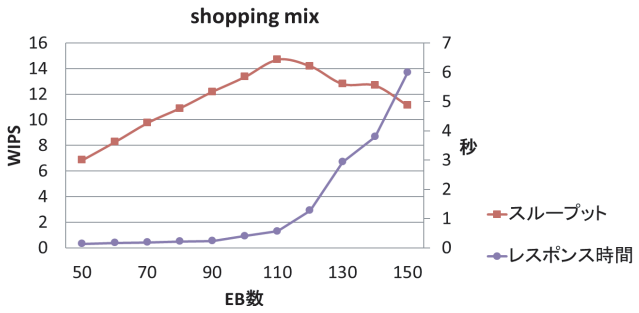

図 4　ローカル環境における shopping mix の性能

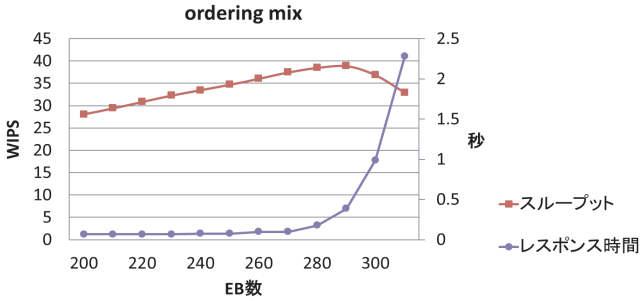

図 5　ローカル環境における ordering mix の性能

5.　広域ネットワーク環境における性能評価

パブリッククラウドは遠隔地にあることを想定し，Dummynet を使って遅延を入れた場合の測定を行った．近隣の街にバックアップを置く場合と，海外のような遠隔にバックアップを置く場合を想定して RTT16ms と 256ms で評価した．3 種のワークロードそれぞれの結果とローカル環境の結果との比較を図 6，7，8 に示す．遅延を入れた場合においても，スループットとレスポンス時間の変化はローカル環境と同じく，EB 数を増やしていくにつれスループットも大きくなり，レスポンス時間は指数関数的に上昇する．ある点において最大のスループットとなり，その後は EB 数を増やしても下降していく． browsing mix と ordering mix では，図 6，7 からも分かる通り，スループット，レスポンス時間ともに遅延による影響はほとんどみられなかった．

一方で，ordering mix では，RTT256ms の場合に遅延による影響がみられた．このときの最大スループットは，ローカル環境では EB 数 290 のとき 38.9WIPS であったのに対

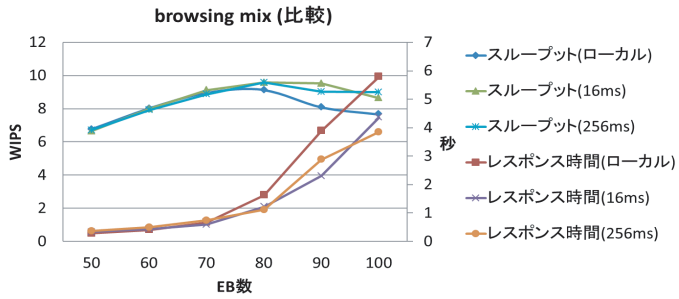

図 6　browsing mix の性能比較

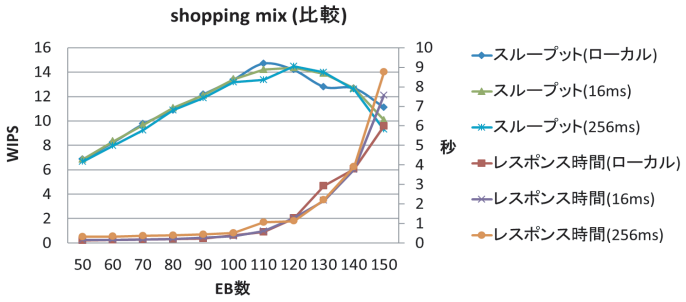

図 7　shopping mix の性能比較

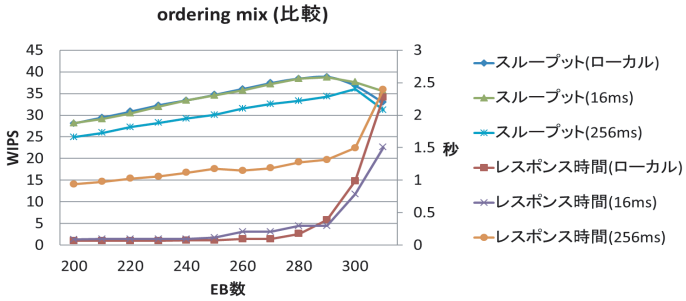

図 8　ordering mix の性能比較

して，EB 数 300 のとき 36.1WIPS であり，およそ 7.2 %下回る．RTT16ms の場合は，ローカル環境とスループット，レスポンス時間ともにほとんど差がなく，性能の低下はみられなかった．RTT256ms の場合は，ローカル環境と比較するとスループットが下回り，レスポンス時間がほとんどのケースで 1 秒を超えていることから，高遅延環境では性能低下が無視できない．ordering mix は更新クエリの割合が 1 番多いため (表 2) Pangea の更新クエリの扱いが原因で遅延による性能低下がみられたと考えられる．

6.　Pangea の修正提案

6.1　更新クエリ修正提案

広域ネットワーク環境での性能向上を図るために，Pangea の修正を検討する．本節では更新クエリ処理の修正を提案する．修正前の Pangea の更新クエリのプロトコルを表 3 に示した．第 2 節でも示した通り，このプロトコルの場合，遠隔にあるデータベースからの応答を待つ時間がかかってしまうために，性能低下がみられると考える．そのため，全レプリカからの応答を待たず，Leader から応答が帰ってきた時点でクライアントに応答を返し，その後に一貫性を保つために Follower に対しても同様に処理を行うように修正する．修正プロトコルを表 4 に示す．

表 3　更新クエリプロトコル

1	クライアントから Pangea にリクエスト送信
2	Pangea から Leader にリクエスト送信
3	Leader から応答が返る
4	全 Follower にリクエスト送信
5	全 Follower から応答が返る
6	Pangea からクライアントに応答が返る

表 4　更新クエリプロトコル修正

1	クライアントから Pangea にリクエスト送信
2	Pangea から Leader にリクエスト送信
3	Leader から応答が返る
4	Pangea からクライアントに応答が返る
5	全 Follower にリクエスト送信
6	全 Follower から応答が返る

修正プロトコルの場合，クライアントは応答を受け取ると次のクエリを送るため，Follower に対するクエリが前のクエリを追い越して順番が前後になると，コンシステンシが崩れてしまうことが懸念される．そこで，次のクエリが追い越さないように，Pangea は全 Follower からの応答を受け取ってから次のクエリを受信し，処理をすることとする．

修正した Pangea での測定結果と修正前の結果とを比較したものを図 9 に示す．遅延による影響が見られた ordering mix における RTT256ms での測定を行った．本節で提案した修正により，クライアントに早く応答が返るためレスポンス時間が短くなると予想していたが，差がほとんどみられなかった．これは，クライアントに応答を即座に返しても Pangea が次のクエリを処理をするのは Follower からの応答を待ってからであるため，更新クエリの応答を早く返したとしても，レスポンス時間に関して改善がみられなかったためであると考えられる．

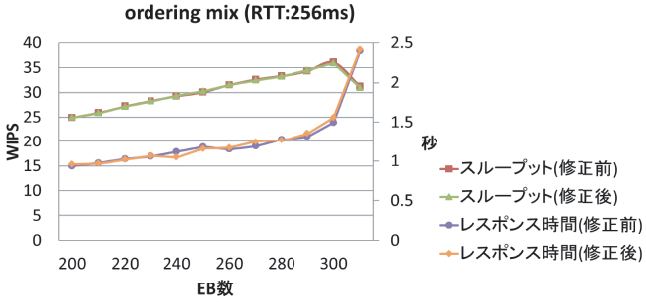

図 9　更新クエリ修正後の ordering mix の性能

6.2 更新クエリの分散処理方式の提案

前節での結果より，Follower に対する処理を行うスレッドを新しく作成し，

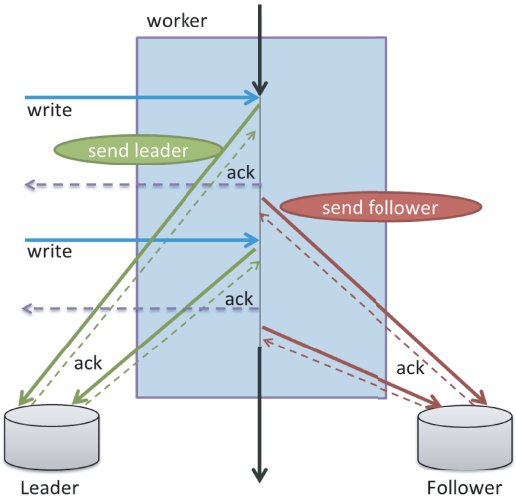

図 10 worker スレッド

従来，Pangea は 1 つのスレッドで処理をしており，全 Follower からの応答を受けてから次のリクエストを送るため，図 10 に示す通り，スレッド worker が直列に処理を行っている．また本研究で行った実験では，Follower は遠隔にあることを想定しているために，前節で提案したように即座にクライアントに応答を返したとしても，次のリクエストを送る前に Follower からの応答を待っている時間が長くなり，タイムロスが発生してしまう．

そこで新たなスレッド，helper を作成し，そのスレッドに Follower に対する処理を投げることにより，Leader に対する処理と Follower に対する処理をスレッド worker と helper が並列に行うような，更新クエリの分散処理方式を提案する．これを図 11 に示す．

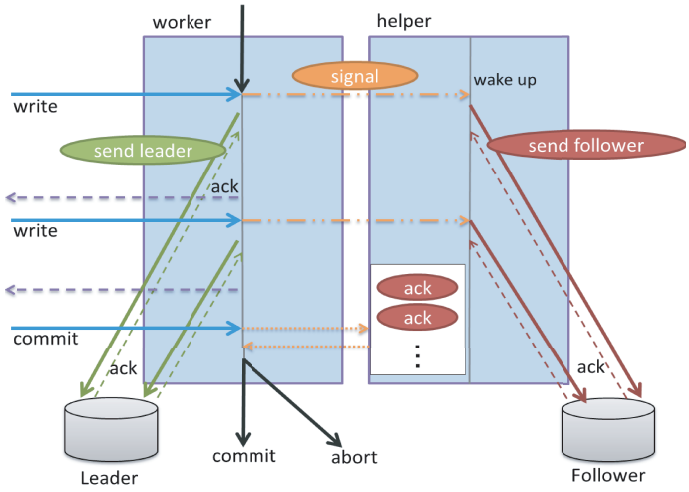

図 11 helper スレッド

helper はスレッド作成後 sleep 状態にしておく．worker は更新クエリを受け取ると Leader にリクエストを送信する一方で，helper に signal を送り呼び起こす．呼び起こされた helper は Follower にリクエストを送り，応答を受け取る．

commit する直前に，worker は helper の Follower からの応答を参照し，成功していれば成功と判断し，失敗あるいは答えが返っていない場合には失敗と判断する．成功の場合は commit し，失敗の場合は abort してトランザクション全体を戻すことでデータベースの一貫性を保つ．

別の 2 つのスレッドに処理を分散させることで並列に処理ができ，従来の方式よりもタイムロスが軽減され，レスポンス時間が短くなることが期待できる．

7. まとめと今後の課題

データベース同期システム Pangea についての基本性能評価を行った．TPC-W ベンチマークの 3 種のワークロードにおけるローカル環境での性能を測定し，EB 数が増えればスループットも上がることが確認できた．EB 数が多すぎればリクエスト数も膨大になり飽和状態になるため，最大のスループットを超えた後に下降することは，妥当だといえる．

パブリッククラウドは遠隔にあることを想定し，遅延をいれた場合の測定も行った．広域ネットワーク環境の場合もローカル環境の場合と同じく，ある点で最大スループットの値になり，その後は下降していくことが確認できた．browsing mix と shopping mix では遅延による性能低下がみられなかったが，ordering mix の高遅延の場合にスループット，レスポンス時間の性能低下がみられた．全てのワークロードにおいて低遅延の場合に性能低下がほとんど見られなかったことから，Pangea は近隣の街など近くにバックアップを置く場合には有益であることが示せた．

広域ネットワーク環境において Pangea の性能向上を図るため，更新クエリに対する修正を提案した．この提案手法では性能向上が見られなかったため，その結果から，更新クエリに対して新たなスレッドを作成し，処理を分散させる方式を提案した．

今後の課題としては，Pangea の更なる性能向上を目指し，今回提案した 2 つのスレッドを使用した分散処理方式の実装を行い，性能評価，結果の考察を行いたい．

参考文献

[1] T.Mishima and H.Nakamura,”Pangea:An Eager Database Replication Middleware guaranteeing Snapshot Isolation without Modification of Database Servers”, Proc.VLDB2009, pp.1066-1077, August 2009. PVLDB2009.

[2] TPC-W http://www.tpc.org/tpcw