

データベース移動に基づく分散データベースシステムにおける 並行処理制御機構

秋山 豊和[†] 酒井 仁^{††}
原 隆浩^{†††} 西尾 章治郎^{†††}

近年、ネットワークの帯域幅が拡大し、分散システムでは、データの転送遅延よりデータの伝播遅延が処理時間に大きな影響を与えるようになってきている。筆者らの研究グループでは、通信回数を削減し、伝播遅延の影響を小さくするため、データベースを移動してトランザクション処理を実行する分散データベースシステムを設計し、そのプロトタイプシステム DB-MAN α を実装している。本稿では、データベース移動がある期間データベースを専有することを考慮し、データベース移動とデータベース操作の同時実行を制御する機構を提案する。さらに、提案した機構を DB-MAN α システムに実装し、その性能を実測評価する。

A Concurrency Control Mechanism in a Distributed Database System Based on Database Migration

TOYOKAZU AKIYAMA,[†] SHINOBU SAKAI,^{††} TAKAHIRO HARA^{†††}
and SHOJIRO NISHIO^{†††}

Due to the recent expansion of the network bandwidth, the data propagation delay has become a significant factor which affects the system performance in place of the data transmission delay. Based on this fact, we have proposed a distributed database system based on database migration, which is a new technology to reduce the impact of propagation delay by dynamically relocating the database through networks. Furthermore, we have implemented a prototype system, DB-MAN α . In this paper, we propose a concurrency control mechanism for DB-MAN α , which allows the parallel execution of database migration and the other database operations to improve system performance. We also show the experiment results regarding the performance evaluation of the proposed mechanism implemented on DB-MAN α .

1. はじめに

近年、ATM(Asynchronous Transfer Mode : 非同期転送モード)方式やギガビット・イーサネットを中心としてネットワーク技術が急速に発展し、それともなあってデータ通信に用いることのできるネットワークの帯域幅も急速に拡大している。従来の分散処理に関する研究では、一度に転送するデータの量を削減することで処理の高速化を図っていたが、ネットワークの帯域幅が急速に拡大されつつある今日では、広い帯

域幅を有効に利用して一度により多くのデータをまとめて転送し、通信の回数を削減することで処理を高速化できる。

従来の分散データベースシステムでは、転送するデータ量の削減が性能向上の重要な要因であったため、データベースを特定のサイトに固定し、処理依頼と処理結果のメッセージ交換によって処理を行っていた。これに対して、広帯域ネットワークを利用すればデータベース全体の転送も短時間で行えるため、データベースを移動してそれに対する処理をローカルに行うことで通信の回数を減らすという手法が可能となる。筆者らの研究グループでは、このようなデータベース自体の移動(データベース移動)をトランザクション処理に用いる手法を提案した⁷⁾。本研究では、数 TB にも及ぶ大規模なデータベースではなく、企業における顧客データや売上データなどのようにテキストベースの比較的小規模なデータベースを対象としている。

[†] 大阪大学サイバーメディアセンター
Cybermedia Center, Osaka University

^{††} キヤノン株式会社情報通信システム本部
Information & Communication Systems Headquarters,
CANON INC.

^{†††} 大阪大学大学院工学研究科情報システム工学専攻
Department of Information Systems Engineering,
Graduate School of Engineering, Osaka University

さらに、筆者らの研究グループは文献 8) において、移動機能を有する分散データベースシステム DB-MAN を提案し、文献 9) においてそのプロトタイプシステムとして DB-MAN α を実装した。DB-MAN α システムでは、データベース移動を用いた処理（移動処理）が従来のデータベース固定型の処理（固定処理）に比べてつねに処理時間が短いとは限らないことを考慮して筆者らが提案した、固定処理と移動処理を適応的に選択する手法¹⁾を用いてトランザクションを実行する。また、DB-MAN α システムではデータベースアクセスを高速化し、データベース移動の効果を高めるため、主記憶データベース^{4),5)}を用いている。さらに、主記憶データベースを用いることで必要となるバックアップについても、移動を考慮した管理機構を実現した。

文献 9) で実装したシステムでは、データベース操作間の同時実行制御は通常の 2 相施錠規約^{2),3)}を用いて実装しているが、データベース移動とデータベース操作の同時実行制御は行っていない。しかし、データベース移動がある期間データベースを専有する処理であることを考慮すると、移動中のデータベースに対するデータベース操作のサポートは処理スループット向上のために重要と考えられる。そこで本稿では、データベース移動に対する同時実行制御機構を設計し、DB-MAN α システムに実装する。

同時実行制御機構は、通常の 2 相施錠規約で用いるデータベース単位の読み出し施錠および書き込み施錠に、データベース移動のための移動施錠の概念を新たに加えることで実現する。読み出し施錠、書き込み施錠と移動施錠の両立性について 3 方式を考案し、それらを DB-MAN α システムに実装する。さらに、実測評価によって実装した 3 方式の性能を比較する。

以下では、2 章でデータベース移動に基づく分散データベースシステム DB-MAN α の概要を述べる。さらに 3 章で同時実行制御機構の設計と実装について述べ、4 章で実装した機構の評価を行う。最後に 5 章で本稿のまとめと今後の課題について述べる。

2. DB-MAN α システムの概要

本章では、まず文献 7) で提案したデータベース移動を用いたトランザクション処理について述べ、次に文献 9) で実装した DB-MAN α システムの概要について述べる。

2.1 データベース移動を用いたトランザクション処理

広帯域ネットワークでは転送遅延が小さいため、大

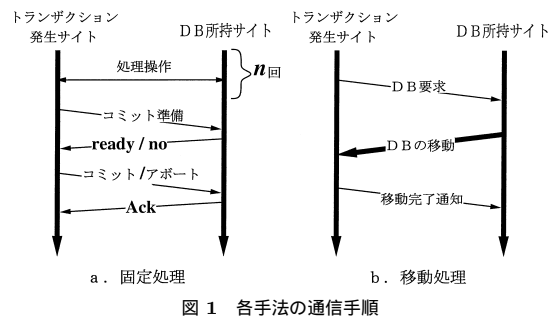


図 1 各手法の通信手順

Fig. 1 Communication process of each processing method.

量のデータ転送は短時間でできるが、伝播遅延は従来のネットワークとほとんど差がないことから、通信の回数を削減することが処理速度向上のために重要である。データベース移動を用いたトランザクション処理（移動処理）では、処理を実行するサイトへデータベース自体を移動することで通信回数を削減し、処理を高速化する。

従来のデータベース固定型の処理（固定処理）では、図 1a に示すような通信手順でトランザクション処理が実行される。まず、トランザクション発生サイトからデータベースを所持するサイトへコネクションが設定される。その後、処理依頼と処理結果などのメッセージ交換が必要な回数だけ行われ、最後に 2 相コミットプロトコル（2PC）に必要なメッセージ交換が 2 往復行われる。これに対し移動処理は、図 1b に示すように、データベース要求、データベース移動、および移動完了通知の 3 回の通信で実現できる。なお、移動元サイトでは、移動の完了後に自サイトのデータベースを削除する。

このようなデータベース移動を用いた処理は、たとえば次のような場合に有効であると考えられる。

- データベースを集中的にアクセスするサイトが時間とともに順次変移していくような場合：

グローバルなデータ共有を考えた場合、各サイトでのアクセスパターンは一樣であったとしても、サイト間の時差などによって順次、集中的にアクセスするサイトが変わっていくことになる。このような場合、集中的にアクセスするサイトの変化に合わせてデータベースを移動していけば通信コストを大幅に削減できる。

- 各サイトに分割して管理しているデータの統計処理を特定のサイトで実行するような場合：

企業における各支社の顧客データおよび売り上げデータについて本社で統計処理を行い、次の戦略を立てる場合が考えられる。統計処理時は多くの

データがやりとりされることになるため、データベースをいったん処理サイトに移動し、処理終了後は元のサイトに戻すようにすれば、通信コストを削減できる。

2.2 DB-MAN α システム

DB-MAN α システムでは、文献 1) の手法に基づいて、トランザクション開始時に直前までのアクセス情報からアクセスの偏りを検出し、固定処理と移動処理を適応的に選択して処理を行う。また、DB-MAN α システムでは、データベース移動の高速化のために、主記憶データベースを用いているため、移動を考慮したバックアップ管理を行う。DB-MAN α システムではテーブルを移動の単位として実装しており、以下ではテーブルをデータベースと呼ぶことにする。

図 2 に DB-MAN α システムのシステム構成を示す。以下では、図中のそれぞれのモジュールの機能について簡単に説明する。これらのモジュールのうち、手法選択部とバックアップ管理部以外は従来の分散データベースとほぼ同様のものである。

インタフェース部： クライアントからトランザクションを受け取り、トランザクション処理手法選択部に問合せを 1 つずつ渡す。一方、部分問合せや移動要求など、他サイトのシステムから受け取った処理要求は、直接トランザクション処理部に渡す。

トランザクション処理手法選択部： トランザクション処理手法選択部は、次の 3 つの各部からなる。

解析部： クライアントから文字列で渡された問合せを解析し、問合せ木を構成して最適化部に渡す。

最適化部： 問合せ木を、実際のデータベース操作を記述したプラン木に変換し、トランザクション処理部へ渡す。

手法選択部： 文献 1) で提案した手法に基づいて、固定処理と移動処理のどちらで処理を行うかを決定する。

トランザクション処理部： トランザクション処理部は、次の 4 つの各部からなる。

トランザクション実行部： トランザクション処理手法選択部から、プラン木と処理手法の情報を受け取り、指定された処理手法でプラン木を実行する。

システムコマンド実行部： 自サイトおよび他サイトのトランザクション実行部およびシステムコマンド実行部から、データベース移動、部分問合せの実行、カタログ情報の更新などのシステムによる処理要求(システムコマンド)を受け取り、実

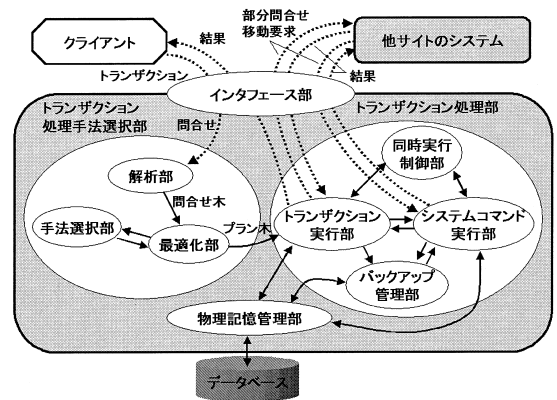


図 2 システム構成

Fig. 2 System composition.

行する。ただし、自サイトからの要求は関数呼び出しによって行う。

同時実行制御部： データベース単位の 2 相施錠規約に基づいて、複数のトランザクションまたはシステムコマンドの同時実行を制御する。

バックアップ管理部： トランザクション実行部から更新操作の内容を受け取り、更新ログを作成して物理記憶に格納する。また、バックアップの送信要求を受けたときには、バックアップと更新ログの内容から最新のバックアップを作成して送信する。

物理記憶管理部： トランザクション処理部からの要求に応じて、物理記憶に対しデータの入出力を行う。

DB-MAN α システムは、カリフォルニア大学バークレー校で開発されたアカデミックフリーのリレーショナルデータベースシステムである POSTGRES^{(11),(12)} をベースに、C 言語で実装した。また、POSTGRES は分散データベースシステムでないため、Mariposa⁽¹⁰⁾ のコードを一部流用してシステムを分散化した。データベースの位置管理方法としては文献 6) において提案しているが、DB-MAN α システムでは位置情報はすべてのサイトで保持し、データベース移動時にはすべてのサイトに通知することにした。

3. データベース移動を考慮した同時実行制御機構

DB-MAN α システムでは、データベース単位の 2 相施錠規約に基づいて、データベース操作の同時実行制御を行う。しかし、文献 9) の実装では、データベース移動とデータベース操作との同時実行制御を考慮しておらず、データベースの移動中に到着した問合せや、

問合せ実行中に到着した移動要求に対応できなかった。

本章では、データベース移動とデータベース操作の同時実行制御機構を設計し、それを DB-MAN α システムに実装する。

3.1 同時実行制御機構の設計

DB-MAN α システムは、2相施錠規約に従う読み出し施錠と書き込み施錠の2種類の施錠を用いてデータベース操作の同時実行を制御する。読み出し施錠としては両立が可能であり、読み出し施錠と書き込み施錠、および、書き込み施錠としてはいずれも両立が不可能である。また、これらの施錠はデータベースを所持するサイトでローカルに管理する。

ここで、データベース移動とデータベース操作の同時実行制御を実現するために、これら2種類の施錠に、新たに移動施錠を導入する。移動元および移動先のサイトは、データベース移動を実行している間、移動中のデータベースに対して移動施錠を確保する。具体的には、移動施錠は、データベースの最初のページの送信を開始する直前に確保され、最後のページの送信を終了して、移動完了を示す確認応答が返された時点で解放される。したがって、移動施錠は2相施錠規約には従わない。

導入した移動施錠と、読み出し施錠および書き込み施錠との両立性には、データベース移動とデータベース操作の同時実行による並行処理性能の向上を重視するか、処理の簡素化によるオーバーヘッドの低減を重視するかで、いくつかのケースが考えられる。そこで、移動施錠と読み出し施錠、書き込み施錠の両立性について、次の3方式を提案する。ただし、どの方式でも移動施錠としては両立せず、後に到着した移動要求を発生したトランザクションは、固定処理に切り替えて実行される。

排他制御方式：読み出し施錠または書き込み施錠と移動施錠は両立しない。

読み出し施錠または書き込み施錠が確保されているデータベースに対して、移動施錠要求が到着した場合は、現在確保されている施錠および先に要求されていたすべての施錠が解放されるのを待って移動施錠が確保される。移動施錠が要求されているデータベースに対して、読み出し施錠または書き込み施錠の要求が到着した場合は、その要求をデータベースの移動先へフォワードする。

並行処理方式：読み出し施錠または書き込み施錠と移動施錠は、移動施錠が後に到着した場合に限り両立する。

読み出し施錠または書き込み施錠が確保されているデータベースに対して、移動施錠要求が到着した場合は、先に確保されていた施錠が解放されるのを待たずに移動施錠を確保できる。ただし、移動施錠を確保する前に読み出しまたは書き込み施錠を要求していたトランザクションは、そのまま移動元のサイトで処理を継続する。移動施錠が確保されているデータベースに対して、読み出し施錠または書き込み施錠の要求が到着した場合は、その要求をデータベースの移動先へフォワードする。

他のトランザクションが読み出しまたは書き込みを行っているデータベースを移動するときは、移動が完了しても移動元のデータベースをすぐには削除せず、読み出しまたは書き込みを行っていたトランザクションが施錠を解放した後に削除する。このとき、移動したデータベースに対して書き込みが行われていたら、書き込み操作のログを移動先へ送信して移動先のデータベースに反映する。移動先では、それまでデータベースに対する操作を禁止する。なお、DB-MAN α システムにおけるログ管理方法は POSTGRES¹¹⁾ のログ管理方法をベースにしており、文献9)において議論しているため、本稿では省略する。

移動中止方式：読み出し施錠または書き込み施錠と移動施錠は両立しない。

読み出し施錠または書き込み施錠が確保されているデータベースに対して、移動施錠要求が到着した場合は、その要求をただちに拒否し、移動施錠を要求したトランザクションを固定処理に切り替えて実行する。移動施錠が確保されているデータベースに対して、読み出し施錠または書き込み施錠の要求が到着した場合は、データベース移動を中止し、移動を行っていたトランザクションを固定処理に切り替えて実行する。

これらの方式の動作例を図3に示す。図では、サイト1において読み出し操作を行うトランザクション1が発生し、続いてサイト2においてデータベース移動後に書き込み操作を行うトランザクション2が発生した場合の、3方式の動作を表している。排他制御方式では、トランザクション1が読み出し施錠を解放するのを待ってデータベースを移動する(図3a)。並行処理方式では、トランザクション1の読み出しと並行してデータベースを移動し、トランザクション1が読み出し施錠を解放するのを待って書き込みを開始する(図3b)。移動中止方式では、データベース移動は失

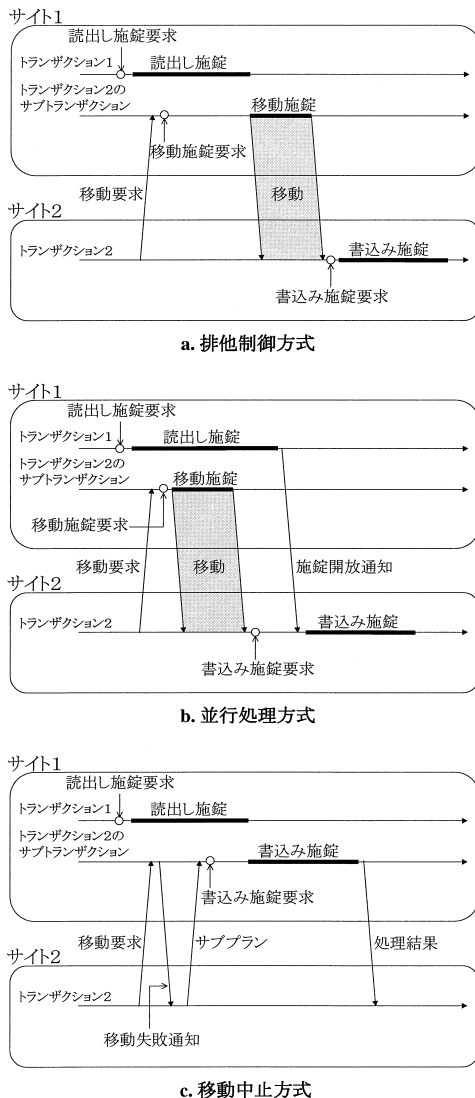


図 3 同時実行制御 3 方式の動作例

Fig. 3 Behavior of the three concurrency control method.

敗し、その後の処理を固定処理に切り替えて実行する (図 3c)。

これらの方式のうち、排他制御方式は最も単純な方式であり、制御用の処理のオーバーヘッドが最も小さい。しかし、先に到着していたトランザクションが施錠を解放するまでデータベースの移動を開始できないため、移動を要求したトランザクションは、施錠の解放を待つ時間と、データベース移動にかかる時間だけ処理遅延が大きくなる。

これに対し、並行処理方式はデータベース移動による遅延の影響を低減できる。この方式でも、先に到着していたトランザクションが施錠を解放するまで、

データベース移動を要求したトランザクションはデータベース操作を開始できない。しかし、データベースの移動は処理中のデータベース操作と並行して実行できるため、移動が完了していれば先に到着していたトランザクションが施錠を解放した時点で移動先のデータベース操作を開始できる。したがって、排他制御方式と比べて、トランザクションの応答時間を短縮できるものと考えられる。しかし、先に到着していたトランザクションが書き込みを行った場合、そのトランザクションの終了時に、更新ログを移動先に送信し、操作内容をデータベースに反映するオーバーヘッドが発生する。

移動中止方式は、移動元サイトでのトランザクションの実行中は、データベースの移動要求を拒否するため、あるサイトに到着したトランザクションが、そのサイトで実行されることを保証する。また、先に到着していたトランザクションが施錠を確保していた場合、移動施錠の要求はただちに拒否されるため、データベースを移動しようとしたトランザクションは、移動施錠が確保できるまで待つことなく処理を継続できる。この方式は、過剰なデータベース移動を防止できる反面、有効な移動を中止してしまう可能性もある。

3.2 同時実行制御機構の実装

3.1 節の提案方式に基づいて、データベース移動とデータベース操作の同時実行制御機構を DB-MAN α システムに実装した。本節では、その具体的な実装方法について説明する。

3.2.1 施錠の管理

DB-MAN α システムは、1 トランザクションを 1 プロセスで実行し、トランザクション間の同時実行制御には、共有メモリを用いる。以下では、あるサイトでローカルに実行されている処理を指す場合、プロセスを用いる。

プロセスが読み出し施錠および書き込み施錠を確保するときは、まず共有メモリ上の施錠情報テーブルを参照し、確保しようとする施錠が他のプロセスの施錠と競合しないかを確認する。競合しない場合は、自プロセスが施錠を確保したことを施錠情報テーブルに書き込む。競合する場合は、共有メモリ上の施錠待ちプロセスキューに自プロセスの ID を書き込み、スリープする。施錠を解放するときは、施錠情報テーブルから自プロセスの施錠に関する情報を削除し、施錠待ちプロセスキューの先頭にあるプロセスにシグナルを送り、スリープから復帰させる。

移動施錠は、施錠情報テーブルと施錠待ちキューおよび位置情報テーブルを用いて管理する。データベー

スを移動するトランザクションは、移動先のサイトで実行しているプロセスによって、移動先のサイトにおける位置情報テーブルのデータベースの現在位置を移動先のサイトに変更し、移動要求フラグを立てる。また、移動元のサイトにおけるプロセスでは、施錠情報テーブルを参照して他のプロセスによる施錠の状態を確認する。移動元サイトで他のプロセスが施錠を確保しているときの動作は、3.1 節で提案した同時実行制御の方式のうち、いずれを用いるかによって異なる。それぞれの方式での動作は次のとおりである。

排他制御方式： 移動要求フラグを立て、施錠待ちプロセスキューに自プロセスの ID を書き込み、スリープする。

並行処理方式： 移動要求フラグを立て、自プロセスが施錠を確保したことを施錠情報テーブルに書き込み、移動を開始する。

移動中止方式： 施錠情報テーブル、施錠待ちプロセスキューおよび位置情報テーブルには変更を加えず、施錠確保に失敗したことを移動を要求してきたサイトへ通知する。

先に施錠が要求されていない場合は、移動中止方式でも移動要求フラグを立てる。他のプロセスが移動要求フラグを先に立てている場合は、同時実行制御の方式にかかわらず、移動施錠の確保に失敗したことを、移動を要求してきたサイトへ通知する。

あるプロセスが読み出し施錠または書き込み施錠を確保するとき、他のプロセスが先に移動要求フラグを立てていた場合、移動中止方式では、3.2.3 項で述べる方法でデータベース移動を中止し、施錠を確保したプロセスの処理を続行する。排他制御方式および並行処理方式では、読み出し施錠または書き込み施錠は確保できず、問合せを 3.2.4 項で述べる方法でデータベースの移動先へフォワードする。

3.2.2 データベース移動とデータベース操作の並行処理

並行処理方式では、他のプロセスが読み出しまたは書き込みを行っているデータベースを移動するために、次のような処理を行う。

- 先に要求されていたすべての施錠が開放されてから、移動元のデータベースを削除する。ただし、移動終了前にトランザクションが終了した場合には、移動の完了まで削除は行わない。
- 書き込み操作を行っているデータベースが他のトランザクションによって移動されたときは、書き込み操作のログを移動先のサイトへ送信する。

これらの処理は、共有メモリ上の位置情報テーブルを用いて、以下の方法で実現した。

- 移動施錠を確保したら、ただちに自サイトの位置情報テーブルを書き換え、データベースの現在位置を移動先のサイトとする。
- 読み出し施錠、書き込み施錠または移動施錠のいずれを解放するときも、施錠待ちキューが空であれば位置情報テーブルを参照し、データベースが移動していないかを確認する。データベースが他サイトに移動している場合は、自サイトにあるデータベースを削除する。

3.2.3 データベース移動の中止

移動中止方式では、移動中のデータベースに対してデータベース操作の施錠を確保するとき、以下の方法で移動を中止する。

- 移動を中止するときは、移動開始時に書き換えられた位置情報テーブルを元に戻し、データベースの現在位置を自サイトとする。
- データベース移動を行っていたプロセスは、移動処理を終了する前に位置情報テーブルを確認する。データベースの位置が移動元サイトになっているときは、移動が中止されているので、移動に失敗したことを移動先のサイトへ通知する。

3.2.4 問合せのフォワード

排他制御方式および並行処理方式において、移動施錠が要求されているときに到着したトランザクション、あるいは、データベースの移動後、位置情報を更新する前に到着したトランザクションは、位置情報テーブルを参照して読み出し施錠または書き込み施錠の要求を、以下の方法でデータベースの移動先へフォワードする。

- データベースの移動元サイトで発生したトランザクションは、他のトランザクションによって移動施錠が先に要求されていたなら、問合せプランを再作成し、サブトランザクションを移動先のサイトで実行する。
- データベースの移動元サイト以外で発生したトランザクションのサブトランザクションは、他のトランザクションによって移動施錠が先に要求されていたなら、問合せの結果を待っているトランザクション発生サイトのプロセスに、データベースの移動先サイトの ID を通知する。通知を受けたプロセスは、自サイトの位置情報テーブルを移動先サイトの ID に書き換え、問合せプランを再作成してサブトランザクションを移動先サイトで実行する。

表 1 実験を行ったシステム環境
Table 1 System environment.

計算機	
機種	Sun Ultra 60 Model 2360
CPU	UltraSPARC-II (360 MHz) × 2
メモリ	256 MB
OS	Solaris 2.6
ネットワーク	
サイト数	3
実効帯域幅	約 80 Mbps
伝播遅延	200 ミリ秒

表 2 実験で用いたパラメータ
Table 2 Parameter configuration.

データベース	
データベース数	1
データベースサイズ	3 ~ 20 MB (10 MB)
トランザクション	
問合せ/トランザクション	5
読み出すデータの量	2 KB
書込むデータの量	2 ~ 32 KB
到着間隔変更周期	20 秒
集中時平均到着間隔	2 秒
散発時平均到着間隔	10 秒

4. 実測評価

本章では、DB-MAN α システムに実装した同時実行制御機構の性能評価のために行った実験とその結果について述べる。

4.1 実験環境

実験では、DB-MAN α システムにおいて、同時実行制御機構の各方式を用いたときのトランザクション処理にかかる平均応答時間を測定した。実験は、トランザクションを読み出しのみとした実験 1 と、書き込み操作を含むものとした実験 2 の 2 通りを行った。それぞれの実験では、比較対象として、すべてのトランザクションをデータベースを固定して処理した場合の平均応答時間も測定した。

実験を行ったシステム環境を表 1 に示す。サイト数は 3 に、データベース数は 1 に固定した。ネットワークは、100 Mbps のイーサネットを用いており、実験では他のトラフィックを流していないため、DB-MAN α システムが利用可能な実効帯域幅は約 80 Mbps である。また、2.1 節で述べたように、移動処理はネットワークの転送遅延より伝播遅延の影響が大きいときに有効となるが、実験を行った環境では想定しているような大きな伝播遅延が生じないため、伝播遅延はプログラム内で擬似的に発生させた。伝播遅延の値は、グローバルにデータ共有を行う場合を考え、日本と米国間程度の遅延を想定し 200 ミリ秒とした。この伝播遅延は、図 1 に示した通信手順に対応して、それぞれのメッセージに対して 200 ミリ秒の遅延を付加した。

実験で用いたパラメータを表 2 に示す。実験 1 ではデータベースのサイズを 3 MB から 20 MB の範囲で変化させることでデータベース移動にかかる時間を変化させた。また、実験 2 ではデータベースのサイズは 10 MB に固定した。データベースのサイズは、適用したアクセスパターンでアクセスした場合に、データベース移動が有効となるサイズを用いた。ネットワークが本研究で想定するように数 Gbps 程度の広帯域な

場合は、データベース移動が有効となるサイズは数十 MB から数百 MB となる。本研究では、数 TB にも及ぶ大規模なデータベースではなく、企業における顧客データや売上データなどのようにテキストベースの比較的小規模なデータベースを対象としている。

提案方式では、問合せの種類が読み出し、書き込みにかかわらず、移動元のトランザクションが終了してから移動先のトランザクションを開始するため、問合せの処理時間が増加した場合の各方式への影響については、読み出し、書き込みによる差異は生じないと考えられる。そこで、並列処理方式のログの転送による影響について調べるため、特に書き込むサイズを変化させた場合のみを示す。書き込みのサイズは問合せの処理時間に差異が見られる程度の大きさで、かつ、同時実行制御以外の部分のシステム性能の影響により、方式間の違いが不明確にならないサイズとして 2 KB ~ 32 KB を指定した。

システムの現段階の実装では、プロセス起動のオーバヘッドなどが大きく、到着間隔が短い場合やサイト数が多くなった場合に同時実行制御以外のオーバヘッドの影響が大きくなり、方式間の違いが不明確になるため、サイト数を 3、集中時の平均到着間隔を 2 秒としている。今後、スレッド化などによる処理の軽量化を行い、より高負荷な環境においても調査する必要がある。

トランザクションは指数分布に基づいて発生させ、ある 1 つのサイトの平均到着間隔を、それ以外のサイトの平均到着間隔に比べて小さな値に設定することで、偏ったアクセスとなるようにした。このようなトランザクションの発生方法は、2.1 節で述べたような、移動処理が有効となるアクセスパターンを想定している。集中してアクセスが発生するサイトの平均到着間隔を集中時平均到着間隔、それ以外のサイトの平均到着間隔を散発時平均到着間隔と呼ぶ。集中的にデータベースにアクセスするサイトは一様分布の乱数によっ

て決定し、到着間隔変更周期ごとにそのサイトを変化させた。

DB-MAN α システムでのトランザクション処理手法選択機構では、特定のサイトからアクセスが集中し、データベースを移動した方が平均処理時間を短縮できると判断したときに、データベースの移動を決定する。しかし、この機構では、過去の履歴に基づいてデータベースを移動するため、有効でない移動が生じる可能性がある。本実験では、このような有効でない移動によって同時実行制御の各方式間の違いが不明確になることを防ぐため、トランザクション処理手法選択機構を用いず、上述の方法で生成した系列をいったん記録して、あらかじめ最適なタイミングを調べたうえでデータベースを移動させた。

4.2 実験 1：読み出しのみのトランザクション

トランザクションを読み出しのみとし、データベースのサイズを変化させたときの各方式の平均応答時間を図 4 に示す。

この実験では、トランザクションは書き込み操作を行わないため、移動したデータベースに対する更新操作のログを送信する処理のオーバーヘッドが生じないことから、並行処理方式が最も良い結果を示した。ただし、データベースサイズが大きくなると、移動処理の有効性が低くなるためにデータベース移動の回数が減少することから、平均応答時間は固定処理のみの場合と近くなった。

排他制御方式では、3.1 節で述べたように、データベース移動にかかる時間だけ処理が遅れるため、データベースサイズの増加とともに並行処理方式との差が大きくなった。データベースサイズが 15 MB のときに並行処理方式との差が小さくなっているのは、データベースサイズの増加とともにデータベース移動の回数が減少したためである。

実験では、平均応答時間が最も短くなるようにデータベースを移動するタイミングを指定してあるため、移動中止方式では、有効なデータベース移動の中止によって、平均応答時間が長くなった。特に、データベースサイズが増加してデータベース移動にかかる時間が大きくなるほど、データベース移動の中止が発生しやすい。この実験では、データベースサイズが 5 MB 以上のときはデータベース移動がすべて中止され、固定処理のみを行った場合と近い結果を示した。しかし、データベースサイズが大きいと、データベース移動の中止処理にかかるオーバーヘッドが大きくなるため、固定処理のみの場合より平均応答時間が長くなった。

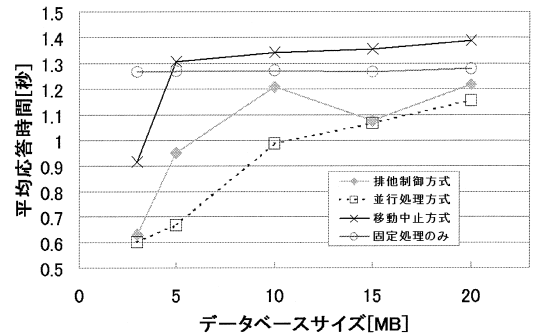


図 4 トランザクションが読み出しのみのとき

Fig. 4 Average response time of read-only transactions.

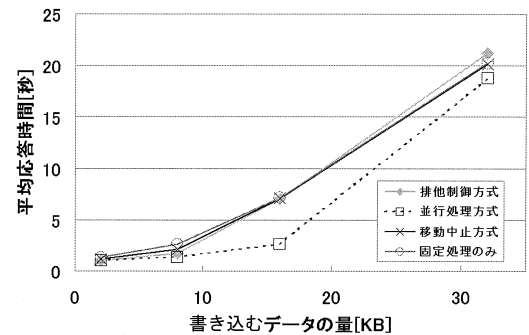


図 5 トランザクションが書き込みを含むとき

Fig. 5 Average response time of transactions with write operations.

4.3 実験 2：書き込みを含むトランザクション

トランザクションが書き込み操作を含むものとし、書き込むデータの量を変化させたときの各方式の平均応答時間を図 5 に示す。

この実験では、書き込むデータの量にかかわらず、並行処理方式が最も短い応答時間を示した。これは、移動するデータベースに対して書き込み操作が行われていて、移動先サイトへ書き込み操作のログを送信する必要があるときでも、その処理のオーバーヘッドは小さいことを示している。

移動中止方式では、書き込むデータの量が増加して書き込み操作を含むトランザクションの処理時間が長くなると、データベース移動を中止する確率が高くなり、固定処理のみの場合と近い結果を示した。特に、書き込むデータの量が 16 KB 以上のときはすべてのデータベース移動が中止され、固定処理のみの場合とほぼ同じ応答時間を示した。

排他制御方式は、書き込み操作を含むトランザクションが終了するまで待ってデータベースを移動するため、データベース移動にかかる時間が直接平均応答時間に影響する。この実験では、データベース移動に

よる処理時間短縮の効果と、データベース移動にかかる時間の影響が打ち消し合い、固定処理のみの場合と近い結果を示した。

5. おわりに

本稿では、DB-MAN α システムにおいて、データベース移動とデータベース操作の同時実行を制御する機構を設計し実装した。同時実行制御機構は、通常のデータベース単位の施錠機構に、移動施錠を新たに導入することで実現した。

移動施錠と、従来の読み出し施錠または書き込み施錠との両立性の異なる3方式を提案し、DB-MAN α システムに実装した。実測評価の結果、トランザクションが書き込み操作を含んでも含まなくても、並行処理方式が最も良い性能を示すことが分かった。

今後の課題としては、データベースの複製を利用してトランザクションのスループットをさらに向上させることが考えられる。また、現在の実装では読み出し、書き込み操作の施錠をデータベース単位としているが、今後はタプル単位など、より細かい粒度の施錠についても導入し、提案した並列処理制御機構に与える影響を調べる必要がある。さらに、他のアプリケーションによるクライアント端末の要求帯域の増加や、複数のデータベースに対するデータベース移動の同時実行により、帯域が不足する可能性があるため、ネットワークの空き帯域の把握や複数のデータベース移動間のスケジューリングについても考慮する必要がある。

謝辞 本研究は、日本学術振興会未来開拓学術研究推進事業における研究プロジェクト「マルチメディア・コンテンツの高次処理の研究」(Project No.JSPS-RFTF97P00501)、および、文部科学省特定領域研究(C)(13224064)の研究助成によるものである。

参考文献

- 1) 秋山豊和, 原 隆浩, 春本 要, 塚本昌彦, 西尾章治郎: アクセス情報に基づくデータベース移動を用いたデータベース再配置手法, 情報処理学会論文誌, Vol.40, No.6, pp.2765-2775 (1999).
- 2) Cellary, W., Gelenbe, E. and Morzy, T.: *Concurrency Control in Distributed Database Systems*, Studies in Computer Science and Artificial Intelligence, Elsevier Science Publishers B.V., Amsterdam (1988).
- 3) Ceri, S. and Pelagatti, G.: *DISTRIBUTED DATABASES Principles and Systems*, McGraw-Hill Computer Science Series, McGraw-Hill Book Company, Singapore, International student edition (1985).

- 4) DeWitt, D., Katz, R., Olken, F., Shapiro, L., Stonebraker, M. and Wood, D.: Implementation techniques for main memory database systems, *Proc. ACM SIGMOD'84*, pp.1-8 (1984).
- 5) Garcia-Molina, H., Lipton, R.J. and Valdes, J.: A massive memory machine, *IEEE Trans. Comput.*, Vol.C-33, pp.391-399 (1984).
- 6) Hara, T., Harumoto, K., Tsukamoto, M. and Nishio, S.: Location Management Methods of Migratory Data Resources in ATM Networks, *Proc. ACM Symposium on Applied Computing (ACM SAC'97)*, pp.123-130 (1997).
- 7) Hara, T., Harumoto, K., Tsukamoto, M. and Nishio, S.: Database migration: A new architecture for transaction processing in broadband networks, *IEEE Trans. Knowledge and Data Eng.*, Vol.10, No.5, pp.839-854 (1998).
- 8) Hara, T., Harumoto, K., Tsukamoto, M. and Nishio, S.: DB-MAN: A distributed database system based on database migration in ATM networks, *Proc. IEEE Data Engineering*, pp.522-531 (1998).
- 9) 酒井 仁, 秋山豊和, 原 隆浩, 春本 要, 塚本昌彦, 西尾章治郎: データベース移動に基づく分散データベースシステム DB-MAN α の設計と実装, 情報処理学会論文誌, Vol.41, No.11, pp.3092-3102 (2000).
- 10) Stonebraker, M., Aoki, P.M., Devine, R., Litwin, W. and Olson, M.: Mariposa: A new architecture for distributed data, *Proc. IEEE Data Engineering*, pp.54-65 (1994).
- 11) Stonebraker, M. and Rowe, L.A.: The design of POSTGRES, *Proc. ACM SIGMOD'86*, pp.340-355 (1986).
- 12) Stonebraker, M., Rowe, L.A. and Hirohama, M.: The implementation of POSTGRES, *IEEE Trans. Knowledge and Data Eng.*, Vol.2, No.1, pp.125-142 (1990).

(平成 13 年 9 月 11 日受付)

(平成 14 年 1 月 8 日採録)

(担当編集委員 掛下 哲郎)



秋山 豊和 (正会員)

1977 年大阪大学工学部情報システム工学科卒業。1999 年同大学院工学研究科博士前期課程修了。2000 年同大学院工学研究科博士後期課程中退後、大阪大学サイバーメディアセンター助手となり、現在に至る。分散処理、データベースに興味を持つ。IEEE、電子情報通信学会の各会員。



酒井 仁

1998 年大阪大学工学部情報システム工学科卒業。2000 年同大学院工学研究科修士課程修了。同年、キヤノン(株)入社、現在に至る。データベースシステムに興味を持つ。



原 隆浩 (正会員)

1995 年大阪大学工学部情報システム工学科卒業。1997 年同大学院工学研究科博士前期課程修了。同年同大学院工学研究科博士後期課程中退後、同大学院工学研究科情報システム工学専攻助手となり、現在に至る。工学博士。1996 年本学会山下記念研究賞受賞。2000 年電気通信普及財団テレコムシステム技術賞受賞。データベースシステム、分散処理に興味を持つ。IEEE、電子情報通信学会の各会員。



西尾章治郎 (正会員)

1975 年京都大学工学部数理工学科卒業。1980 年同大学院工学研究科博士課程修了。工学博士。京都大学工学部助手、大阪大学基礎工学部および情報処理教育センター助教授を経て、1992 年より大阪大学大学院工学研究科情報システム工学専攻教授となり、現在に至る。2000 年より大阪大学サイバーメディアセンター長を併任。この間、カナダ・ウォータールー大学、ピクトリア大学客員。データベース、知識ベース、分散システムの研究に従事。現在、ACM Trans. on Internet Technology, Data & Knowledge Engineering, DataMining and Knowledge Discovery, The VLDB Journal 等の論文誌編集委員。ACM, IEEE 等 8 学会の会員。