ＯＳＩ−ＴＰにおけるコミット制御手順の最適化に関する一考察

2Q−3

中川路哲男、勝山光太郎、水野忠則

三菱電機(株)　情報電子研究所

１．はじめに

ISO及びCCITTでは、異機種のシステム間での相互接続を可能とする為に、ＯＳＩ(開放型システム間相互接続)の標準化を行なっている。ＯＳＩの応用層のサービスの一つに、トランザクション処理(ＴＰ)がある。ＴＰは、ネットワーク上に分散されたシステム間で頻発するデータ通信を、効率良く、かつ安全に遂行するためのサービス・プロトコルである。ＴＰの機能の一つに、分散されたシステムのデータの一貫性を保証するためのコミット制御がある。 1989年にDIS化された基本標準では、その仕様はほぼ固まったものとなっている。

一方、ＴＰの拡張機能として、コミット制御手順の最適化が検討されている。これは、コミット制御に関わる手順を最適化し、トランザクション処理に関する性能向上を目的としたものである。

我々は、ＴＰにおけるコミット制御手順の最適化の特性や効果について考察したので、その結果について報告する。

２．ＴＰにおけるコミット制御手順

ＴＰにおいては、ＣＣＲを利用して、２相コミットの手順によりコミット制御を行なう。これは、コミットの手順を以下の二つの相に分けて行ない、データの一貫性を保証する機構である。

＜第一相＞トランザクションのためのデータ処理を行なった後に、コミットの可否を問い合わせ、結果を集計

＜第二相＞結果に従い、コミット(またはロールバック)を実行

この手順は、トランザクション木と呼ばれるトランザクション処理のために結合されたシステム間の木構造の関係において、下流方向に分配、または上流方向に収集されていく。

コミット制御手順の例を図１に示す。図１において、☆から★までが第一相であり、★から●までが第二相である。なお、図中の矢印は事象の契機であり、必ずしもＴＰサービスとは対応しない。

３．コミット制御手順の最適化

1989年にDIS化された基本標準では、上記のコミット制御手順を規定している。一方、その拡張機能として、コミット制御に関わる手順を最適化し、トランザクション処理に関する性能向上を図る、コミット手順の最適化が検討されている。この最適化のために、様々な機能が案として挙げられているが、本稿では、利用者に提供するサービスを変更することなく最適化を図る機能であるＬＳＯ(Last Subordinate Optimization)を対象とする。即ち、コミット開始前にコミット準備の情報を送る機能はここには含めない。

ＬＳＯは、自分を含めた隣接ノードの一つ以外が、コミット可となった時点で、その隣接ノードにコミット可の情報を通知することができる機能である。これにより、第一相の問い合わせの結果をトランザクション木のルートに集めるのではなく、問い合わせの返答の遅いところに集めることが可能となる。問い合わせの返答のもっとも遅いところが返答した時点でコミット判断を下すことができるため、トランザクションに関わる時間を短縮することができる。

ＬＳＯの例を図２に示す。ルートノードＡは、ノードＢからコミット可の返答を受けた時点で、自分もコミット可であるので、それをノードＣに通知する。ノードＣにノードＤとノードＥからコミット可の返答が到来した時点で、本トランザクションのコミット判断が下され、第二相に入るので、トランザクションに関わる時間が短くて済む。

４．ＬＳＯの特性に関する考察

ＬＳＯの詳細な仕様はまだ規定されていないが、ここではNWI(新作業項目)へ日本から提案した仕様を前提とする。

特性を考察する場合の仮定として、下位層と物理回線における通信・伝送処理と、ログなどの取得のためのディスクアクセスの時間に比して、ソフトウェアの走行時間を十分小さいとした。これにより、コミット制御に要する時間は、以下の処理に要する時間から推定することができる。

・システム間でデータを伝送するための時間＝α：ここでは、全てのシステム間で、また全ての種類のデータで同一と仮定する。

・あるシステムにおいてトランザクションをコミット可の状態にするための時間＝β：この時間には、データベースにおけるコミット準備処理や、通信処理モジュールにおけるログ取得時間が含まれる。

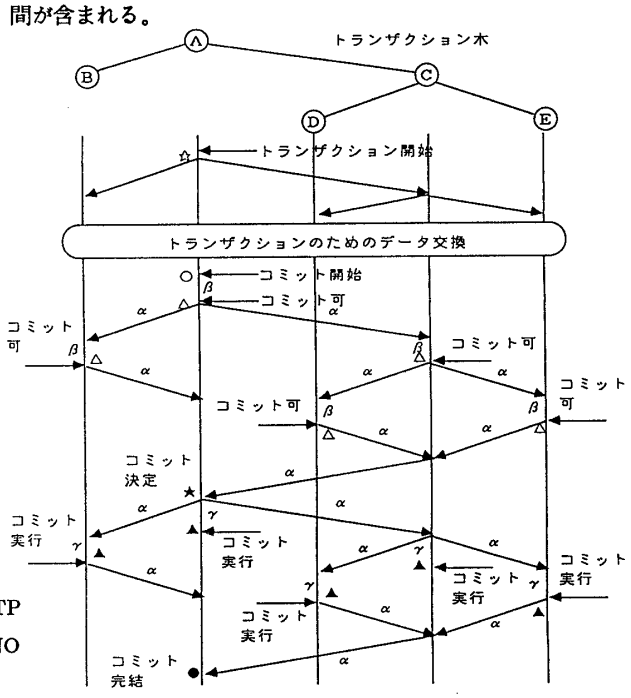

図１　コミットメント制御手順の例（ＬＳＯを使用しない場合）

A study on optimization of the commitment procudure in the OSI TP

Tetsuo NAKAKAWAJI, Kotaro KATSUYAMA , Tadanori MIZUNO

Information Systems and Electronics Development Laboratory
MITSUBISHI ELECTRIC CORPORATION

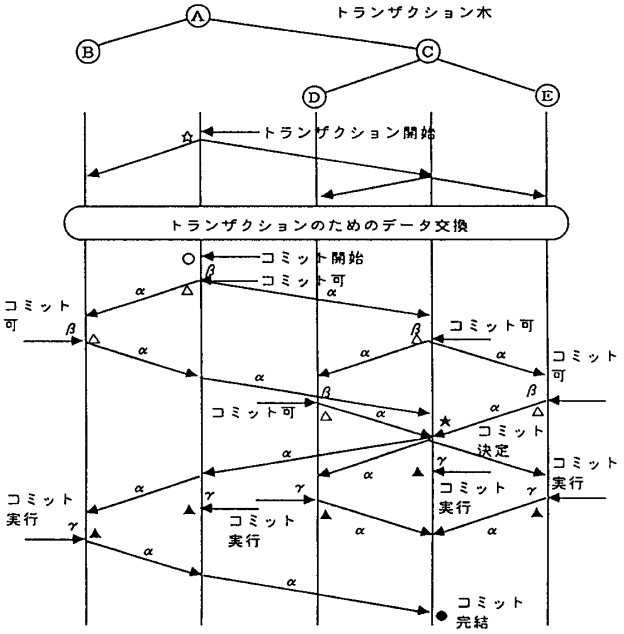

図２　コミットメント制御手順の例（ＬＳＯを使用した場合）

・あるシステムにおいてトランザクションのコミットを実行するための時間＝γ：この時間には、データベースにおける更新イメージの確定処理や、通信処理モジュールにおけるログ消去時間が含まれる。

評価の対象とする処理時間としては、トランザクションのコミットに要する時間と、データベースのロック時間を考えた。データベースのロック時間とは、トランザクションのためにデータベースをロックしてから、コミットを行なってアンロックまでの時間である。この間データベースの更新対象部分は、他のトランザクションからは使用できないため、並行制御の面からはクリティカルな時間ということができる。

また、通常データベースのロックは、更新要求が到来した時点で行なう。しかし、正確にいえばコミット可の応答を返すまでは、デッドロック回避などのためにトランザクションをロールバックしてデータベースをアンロックすることが可能である。そこで、データベースのロック時間を、アンロックできない時間、即ちコミット可の応答を返してからロックを解除するまでの時間として定義することとした。

例えば、図１、図２においては、トランザクションのコミットに要する時間は、○から●まで、データベースのロック時間は各ノードにおける△から▲までである。これを上記各処理時間から求めた結果を表１に示す。ＬＳＯにより、コミットに要する時間がα短縮され、またデータベースのロック時間もノードＤとノードＥでは２α短縮されていることが分かる。

次に、これを一般的なトランザクション木に拡張して、処理時間の推定を行なった。

ＬＳＯを使用する場合は、処理時間はトランザクション木の形状に依存する。定性的には、ルートノードから最後にコミット可となるノードの方に、コミット可の情報が伝われば伝わるほど、最適化を図ることが可能となる。即ち、最も最適化が可能な場合は、図３のような場合である。ＬＳＯを使用する場合は、このように効果が最大になる場合について、求めることとした。

トランザクション木においてルートから最も遠いノードまでのコネクションパスの数をｎとしたときの、両処理時間を計算した結果を表２に示す。これより、以下のことが考察される。

・ＬＳＯの効果は、ルートノードから見てトランザクション木が非対称であればあるほど、遠いノードにコミット可を通知することができるので効果が大きい。

・トランザクションのコミットに要する時間は、ＬＳＯを使用することにより、最大$(4n\alpha+(n+1)\beta+\gamma)$から$(3n\alpha+(n+1)\beta+\gamma)$に短縮することが可能である。例えば$\alpha=\beta=\gamma$とし、ｎがある程度大きいとすると、約２０％の削減効果ということになる。また、低速度の通信回線で$\alpha>\beta=\gamma$とし、ｎがある程度大きいとすると、約２５％の削減効果ということになる。これがＬＳＯで期待できる最大効果である。

・データベースのロック時間も、ＬＳＯで最適化されるが、最長時間は変わらない。

・ルートノードにおけるデータベースのロック時間は、ＬＳＯで最適化されるに従い長くなる可能性がある。よって、ルートノードにおける最大ロック時間$(2n\alpha+n\beta+\gamma)$がそのシステムにとって長過ぎる場合には、コミット可の通知を遅らせるなどのシステム調整が必要となる。

５．おわりに

ＴＰにおけるコミット制御手順の最適化について考察を行なった。今後はこの拡張仕様の標準化に積極的に参加すると共に、実装を通じてその効果を検証していく予定である。

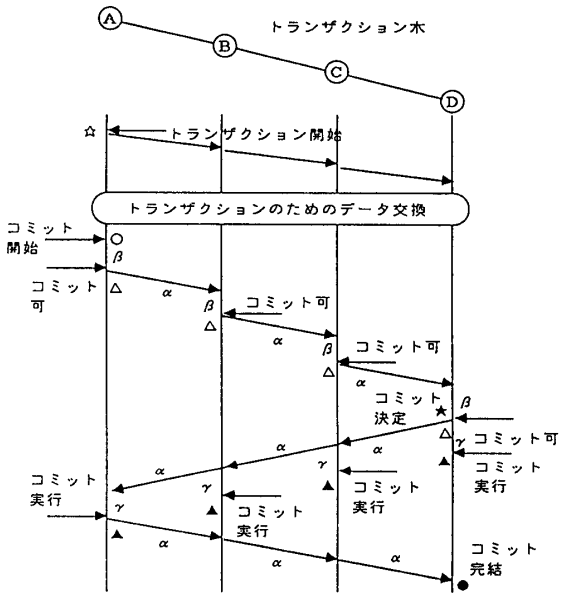

図３　ＬＳＯが最も有効なコミットメント制御手順の例

表１　ＬＳＯ機能の有無による処理時間の比較(図１、２の場合)

処理時間		ＬＳＯなし(図１)	ＬＳＯあり(図２)
コミットに要する時間		$8\alpha+3\beta+\gamma$	$7\alpha+3\beta+\gamma$
データベースロック時間	ノードＡ	$4\alpha+2\beta+\gamma$	$4\alpha+2\beta+\gamma$
	ノードＢ	$4\alpha+\beta+\gamma$	$4\alpha+\beta+\gamma$
	ノードＣ	$4\alpha+\beta+\gamma$	$2\alpha+\beta+\gamma$
	ノードＤ	$4\alpha+\gamma$	$2\alpha+\gamma$
	ノードＥ	$4\alpha+\gamma$	$2\alpha+\gamma$

表２　ＬＳＯ機能の有無による処理時間の比較(一般的な場合)

処理時間	ＬＳＯなし	ＬＳＯあり
コミット可否の問い合わせ時間	$2n\alpha+(n+1)\beta$	$n\alpha+(n+1)\beta$
コミットの実行のための時間	$2n\alpha+\gamma$	$2n\alpha+\gamma$
コミットに要する時間(合計)	$4n\alpha+(n+1)\beta+\gamma$	$3n\alpha+(n+1)\beta+\gamma$
データベースロック時間	$2n\alpha+i\beta+\gamma$	$2i\alpha+j\beta+\gamma$
	$(0\leqq i,\ j\leqq n)$	

ｎ＝ルートノードからの最大パス数