

CCRにおけるロック方法に関する一考察

2Q-4

足立 美壽津 松田 栄之

NTTデータ通信株式会社

1. はじめに

OSIでは、分散処理指向のアプリケーションとして、TP(トランザクション処理)やRDA(遠隔データベースアクセス)の標準化が進められている。これらの分散アプリケーションの複数資源の更新制御のために2相コミットメントを採用したプロトコルとしてCCR(コミットメント、同時性及び回復制御)の標準化が進められている。

CCRを用いたアプリケーションでは、同時に起こるトランザクション間で資源の独立性を保証するために排他制御をする必要がある。排他制御の一つに処理を行う資源を確保(ロック)する方法があるが、ロックを行うタイミングは性能に影響を及ぼすと考えられる。そこで、ロックについて2つの方法を提案し、その特性をシミュレーション解析により求め、分散通信の設計に役立てる事とした。

2. CCRの概要

CCRは複数のノードにある資源を同時に更新する時に、ノード間の処理の同期を取るためのプロトコルであり、資源の一貫性を保証し、障害が発生しても正しい処理の進行を保証するプロトコルである。

CCRは処理を達成するために2相コミットメントを採用している。2相コミットメントは、各資源の状態を問い合わせ(1相目の処理)、資源の更新を処理する(2相目の処理)ことにより分散された資源の同期した更新を保証する。

3. 前提条件

3.1 モデル

ここでは、CCRのサービスをモデル化し、さらに次の仮定を設ける。

- ①トランザクションが発生するノードを受付ノード、更新処理を行う資源のノードを処理ノードと呼ぶ。
- ②トランザクション処理の基本形として、2つの処理ノードを選んで更新する処理を対象とする。処理ノードの選択に関して、それぞれの処理ノードに重みを付ける。
- ③トランザクションの発生時間間隔は指数分布に従う。
- ④ノード間のプロトコル処理を含んだ通信時間はトランザクション、処理ノードを問わず一定とする。また処理ノード毎の処理時間は一定とする。
- ⑤処理ノードでの待ちキューの大きさは無限大とする。ただし、許容待ち時間を越えたトランザクションは実行されない。
- ⑥使用するCCRサービスプリミティブと役割を表1に示す。

3.2 ロック方法とシーケンス

ロック方法1：全体ロック方法

- ①処理ノードはC-BEGINindを受信した時に資源をロックして処

理を開始する。

- ②C-BEGINind到着時に資源が他のトランザクションによってロックされている場合は、許容された時間を待つ。資源が解放されない場合は、C-ROLLBACKreqを発行して、トランザクションを異常終了させる。
- ③2つの処理ノードの上位アプリケーションの処理終了後、資源はコミットされ、トランザクションは終了する。

ロック方法2：2相目ロック方法

- ①処理ノードはC-BEGINindを受信した時に処理を開始する。
 - ②資源がロックされている場合の処理はロック方法1の②に同じ。
 - ③処理中に、あるトランザクションが資源をロックした場合、他のトランザクションにはC-ROLLBACKreqが発行され、異常終了させられる。
 - ④処理が終了したトランザクションは資源をロックし、C-READYreqを発行する。
 - ⑤2つの処理ノードからC-READYindを受信した受付ノードはコミットを命令し、トランザクションを終了する。
- ロック方法2のコミットメントシーケンス例を図1に示す。

3.3 パラメータ

- ①資源数(N)：受付ノードからアクセス可能な資源の数。本考察では $N=8$ としている。
- ②処理時間($\sigma_i(i=1..N)$)：各処理ノードが1つのトランザクションの上位アプリケーションの処理の為に必要とする時間。各処理ノードの処理時間の平均値 σ を基準時間とし、これ以降の時間に関するパラメータは、基準時間の比で表す。
- ③コミット時間(χ)：トランザクションがコミットを命令した時に、処理ノードでコミットメントのために必要な時間。
- ④ノード間通信時間(τ)：受付ノードと処理ノード間のプロトコル処理を含んだ通信時間。
- ⑤許容待ち時間(ω)：C-BEGINindが処理ノードに到着したにも関わらず、資源がロックされているため、資源が解放されるのを待つ許容時間。
- ⑥トランザクション発生数(ρ)：基準時間に発生するトランザクションの数の平均。

表1 CCRサービスプリミティブと役割

名称	処理の内容	本稿で付加した役割
C-BEGIN	アトミックアクション開始	資源をロックする(ロック方法1)
C-PREPARE	コミット可否の問い合わせ	
C-READY	コミット肯定応答	資源をロックする(ロック方法2)
C-COMMIT	コミット要求	
C-ROLLBACK	コミット取消要求	異常終了時、ロックされた資源を解放
C-RECOVER	アトミックアクション回復要求	使用しない

*C-PREPAREに対するコミット可否は、アプリケーションに依存するので、ここでは必ずC-READYを返す。

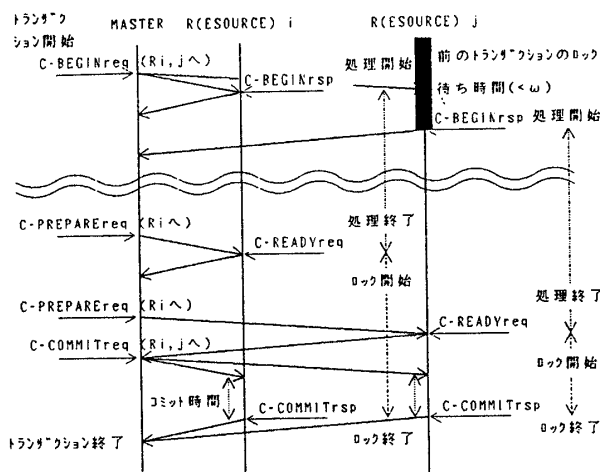


図1 ロック方法2のコミットメントシーケンス例

(受信側は、reqに対してindを、rspに対してcnfを受信する)

A Study for Locking Method using OSI CCR protocol.

Misuzu ADACHI, Shigeyuki MATSUDA,

NTT DATA Communications Systems Corporation

⑦資源のアクセス重み ($\alpha_i(i=1..N)$): 資源*i*を α_i の確率でアクセスする. 本考察では $\alpha_j(j=1)=0.5$, $\alpha_k(k=2..4)=0.1$, $\alpha_l(l=5..8)=0.05$ としている.

3.4 評価尺度

- ①トランザクション成功率 (S): 発生したトランザクションのうち, 正常終了できた割合.
- ②資源のロック率 (L): 最初のトランザクションが開始されてから最後のトランザクションが終了するまでの時間に対する, 各資源がロックされていた時間の比の平均.
- ③アボート率 (A): ロック方法2の場合, 処理を開始したが他のトランザクションのロックによってアボートしたトランザクション数の, 全トランザクション数に対する比.

4. 検証の内容

3.4の評価尺度に従い, 次の検証を行った.

- 検証1: 各ロック方法によるトランザクション成功率の比較
- 検証2: 各ロック方法による資源のロック率の比較
- 検証3: ロック方法2の時のアボート率の傾向

5. 結果と考察

(I) 検証1(図2)

①トランザクション成功率Sは, 許容待ち時間 ω が小さい環境ではロック方法による差はほとんどないが, ω が大きくなるとロック方法1のSが急激に増加する. これは, ロック方法2で待つ事によって処理を開始したトランザクションが, 他のトランザクションのロックによってアボートされるためである. このことから, ロック方法2では ω の効果有余り得られない.

②従ってトランザクションの処理に時間的な余裕があり, ω が大きいような環境ではロック方法1が有効である事がわかる.

(II) 検証2(図3)

①ロック方法の違いにより, 資源のロック率Lの値は大きく違う. これは, ロック方法1では, 上位アプリケーションの処理の間も資源をロックしている為, 一つのトランザクションのロック時間が長いためである.

②ロック方法1では, 許容待ち時間 ω が大きい環境でLが極めて大きい値を示す. これは, ω の効果により資源毎に処理の開始時間が異なり, 他の資源の処理の終了を待つ間もロックされているためである.

③ロック方法2では, コミット時間 ϵ が大きい環境でLが極めて大きい値を示す. これは, ロック方法2では一つのトランザクションのロック時間が ϵ の長さ大きく影響を受けるためである.

④トランザクション処理は, 更新処理だけでなく読みだしの処理も多いと考えられる. 読みだし処理のようにコミットを必要としない処理は多くの同時実行が可能であり, 読みだし処理の多い環境では, Lが全体に小さいロック方法2が有効である事がわかる.

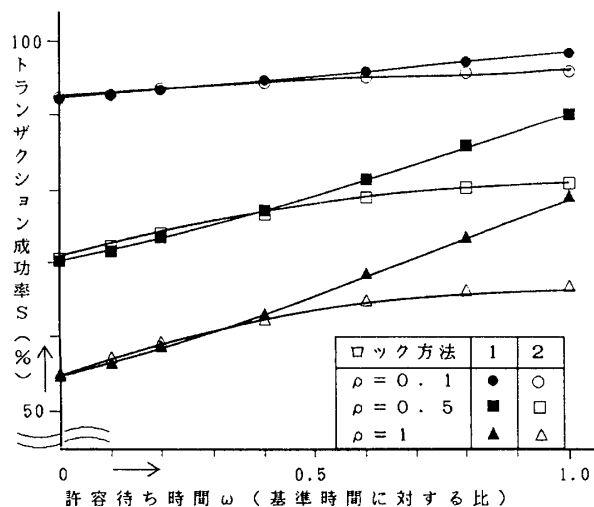


図2 許容待ち時間とトランザクション成功率の関係 ($\epsilon = 0, \tau = 0.1$)

(III) 検証3(図4)

①ロック方法2では, トランザクション発生数 ρ が大きい環境でアボート率Aが大きくなり, 処理を始めたが異常終了してしまうトランザクションが多い事がわかる. これは, ρ が大きい時, 同時に処理されているトランザクションは多いが, 結局その中の一つ(ロックできたトランザクション)のみ処理されるためである.

②またAは, 許容待ち時間 ω が大きい環境で顕著に大きい値を示す. これは, C-BEGINindを処理ノードが受信した時に資源がロックされていても ω の効果によって処理が開始され, 同時実行されるトランザクションが多いためである.

6. まとめ

CCRを用いた時のロック方法について2種類の方法を提案し, その比較を行った. 更新処理が多く, 許容待ち時間の大きい環境ではロック方法1の方が有効であり, 読みだし処理の多い環境では, ロック方法2の方が有効である事がわかった.

参考文献

- (1) ISO/IEC 3RD DIS 9804: Service definition for the Commitment, Concurrency and Recovery(1989)
- (2) 足立, 松田, 岩倉: OSI CCRを用いた同期制御特性に関する一考察, 情報学会第40回全国大会(1990)

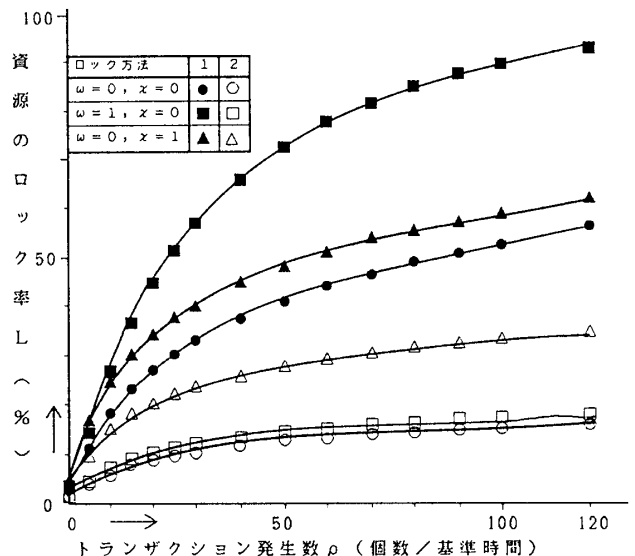


図3 トランザクション発生数と資源のロック率の関係 ($\tau = 0.1$)

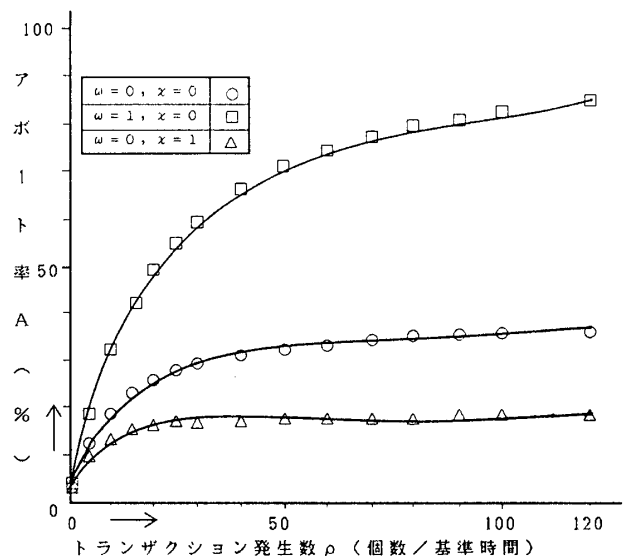


図4 トランザクション発生数とアボート率の関係 ($\tau = 0.1$)