情報処理学会第39回(平成元年後期)全国大会

多版同時実行制御に関する一考察

5M-6

片岡　良治，　武田　英昭，　佐藤　哲司，　井上　潮

ＮＴＴ情報通信処理研究所

１．はじめに

多版同時実行制御は，データベースの更新履歴（版）を参照可能としながら，一貫性のあるトランザクション処理を実現する技法であり，同一のデータに対する参照と更新を同時に行えない単版同時実行制御に比べ，トランザクションの同時実行性を高めることができる．本稿では，これまでに提案されている多版同時実行制御アルゴリズムの問題点を軽減した新しいアルゴリズムを提案し，その効果を示す．

２．従来アルゴリズムとその問題点

これまでに提案されている代表的な多版同時実行制御アルゴリズムとして，二相ロック多版同時実行制御アルゴリズム（ＭＶ２ＰＬ）と基本タイムスタンプ順序多版同時実行制御アルゴリズム（ＭＶＴＯ）がある[1]．各アルゴリズムの概要とその問題点を以下に示す．

（１）ＭＶ２ＰＬ（図１）

［概要］データの更新が完了するまで，更新前の内容を参照可能としながら，二相ロックに基づきトランザクションの一貫性を保証する．単版の二相ロック制御で禁止される同一のデータに対するリードロックとライトロックの同時確保が可能である．ライトロックの解除を契機に参照対象を更新後の内容へ変更するが，更新前の内容を参照中のトランザクションの一貫性を保証するため，この操作はそのデータに確保されているリードロックが解除された後に行う．対象変更が必要以上に遅延されるのを防ぐため，以降，そのデータに対する新たなリードロックの確保を禁止する．

［問題点］データの更新が完了すると，そのデータの参照がリードロックの解除まで待たされる．トランザクションがリードロックを長期間確保する場合，待ちが多発し，デッドロックが起きる可能性が高くなる．

（２）ＭＶＴＯ（図２）

［概要］データの版毎にそれを生成したトランザクションのタイムスタンプ（ライトスタンプ）とそれを参照したトランザクションのタイムスタンプの最大値（リードスタンプ）を付与する．データを参照するときは，トランザクションのタイムスタンプ以下で最大のライトスタンプを持つ版を選択し，トランザクションの一貫性を保証する．条件を満たす版は必ず存在するので，単版の基本タイムスタンプ順序制御と違い，参照が失敗することはない．データの更新は，トランザクションのタイムスタンプが最新版のリードスタンプ以上かつライトスタンプ以上であるときのみ可能である．また，トランザクションの回復可能性を保証するため，更新中の最新版に対する参照，更新は，更新を行っているトランザクションが終了するまで実行待ちとする．

［問題点］データの最新版が頻繁に参照されると，リードスタンプの影響を受けて，更新が失敗する可能性が高くなる．また，データの最新版を参照するとき，それが更新中であると，参照待ちが生じる．

３．協調型多版同時実行制御

上述した問題点をＭＶ２ＰＬとＭＶＴＯを協調させることで軽減するアルゴリズム（ＭＶＨＣ）を提案する．本アルゴリズムの概要を以下に述べる（図３）．

［１］更新を含むトランザクション（更新トランザクション）をＭＶ２ＰＬで管理する．従来のＭＶ２ＰＬと異なり，更新トランザクションが終了するとき，それが生成

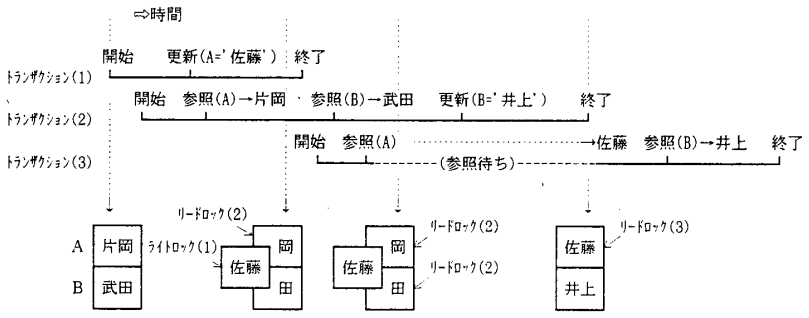

図１　ＭＶ２ＰＬによる制御

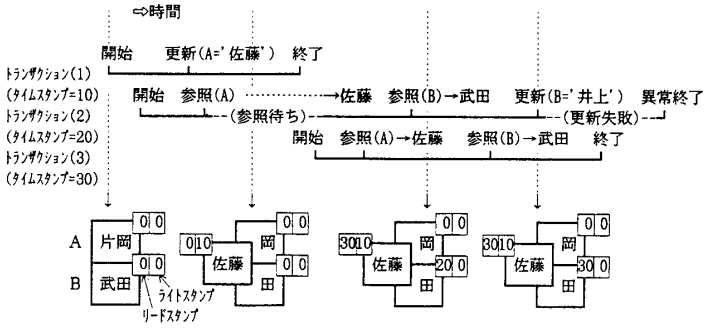

図２　ＭＶＴＯによる制御

A study of multiversion concurrency control algorithms
Ryoji KATAOKA, Hideaki TAKEDA, Tetsuji SATOH, Ushio INOUE
NTT Communications and Information Processing Laboratories

した版に，終了時間に相当するタイムスタンプを付与する．

［２］データの参照のみを行うトランザクション（検索トランザクション）をＭＶＴＯで管理する．版の選択は，［１］で版に付与したスタンプをライトスタンプと考え，ＭＶＴＯの版選択規則に従い行う．従来のＭＶＴＯと異なり，検索トランザクションには，それが発生する時点で版に付与されているライトスタンプの最大値をタイムスタンプとして付与する．また，ＭＶＴＯで更新を扱わなくなるので，版にリードスタンプを付与する必要はない．

本アルゴリズムでは，前節で述べた従来アルゴリズムの問題点を，次のように軽減している．

（１）データベースの全てのデータを参照するような，リードロックを長期間確保するトランザクションをＭＶＴＯで管理し，ＭＶ２ＰＬにおける参照待ち発生を緩和する．

（２）ＭＶＴＯで更新を扱わないこととし，ＭＶＴＯにおけるリードスタンプの影響による更新失敗の発生を解消する．

（３）更新が完了するまで版にライトスタンプを付与しないことで，ＭＶＴＯにおけるデータの最新版に対する参照待ち発生を解消する．

尚，これまでに，更新トランザクションを単版の二相ロック制御で，検索トランザクションを多版の基本タイムスタンプ順序制御で管理する協調型の多版同時実行制御アルゴリズム（ＳＶ２ＰＬ＋ＭＶＴＯ）が提案されている[1]．これでは，トランザクション開始時に一意に定められたデータを参照して更新するような更新トランザクションについて，参照と更新の同時実行性を高める意味がないことから，更新トランザクションを単版で制御している．本稿で提案するアルゴリズムは，更新するデータがトランザクション開始時に定まっておらず，いくつかのデータを参照してそれらの一部を選択的に更新するような更新トランザクションを考慮し，更新トランザクションについても多版制御を採用している点でこれまでのものと異なる．

４．性能評価

上述した４つのアルゴリズムのトランザクション同時実行性を，制御オーバヘッドを全てのアルゴリズムで同一として，シミュレーションにより評価した．評価に用いたトランザクションは，次の２種類である．

①更新トランザクション：データベースを構成する１００個のデータから，１０個をランダムに選択して参照し，その内の１個を更新する．

②検索トランザクション：データベースを構成する１００個のデータ全てを逐次的に参照する．

検索トランザクションの実行頻度を１回／秒に固定し，更新トランザクションの実行頻度を変化させ，更新／検索トランザクションの完了率の変化を求めた．結果を図４に示す．

シミュレーション結果より，以下のことが確認できた．

（１）ＭＶ２ＰＬでは，更新トランザクションの実行頻度が高くなると，検索トランザクションの待ちがデッドロックの原因となりアボートされやすくなるため，検索トランザクションの完了率が悪くなる．

（２）ＭＶＴＯでは，更新トランザクションの実行頻度が高くなると，リードスタンプの影響を受けて更新が失敗することが多くなり，更新トランザクションの完了率がが悪くなる．

（３）ＳＶ２ＰＬ＋ＭＶＴＯは，更新トランザクションを単版で制御する分，ＭＶＨＣに比べ更新トランザクションの完了率が劣る．

（４）ＭＶＨＣは，ＭＶ２ＰＬの更新トランザクションの完了率の高さとＭＶＴＯの検索トランザクションの完了率の高さの両面を備えている．

５．おわりに

本稿では，これまでに提案されている２つの多版同時実行制御アルゴリズムを協調させて，各々の問題点を相互に補完する協調型多版同時実行制御アルゴリズムを提案し，そのトランザクション同時実行性を評価した．本アルゴリズムでは，制御オーバヘッドが他のものに比べ大きくなる可能性が高く，今後，制御オーバヘッドを含めた性能評価を行う予定である．

［参考文献］

[1] P.A.Bernstein, et al.: "Concurrency Control and Recovery in Database Systems", ADDISON-WESLEY PUBLISHING COMPANY, 1987

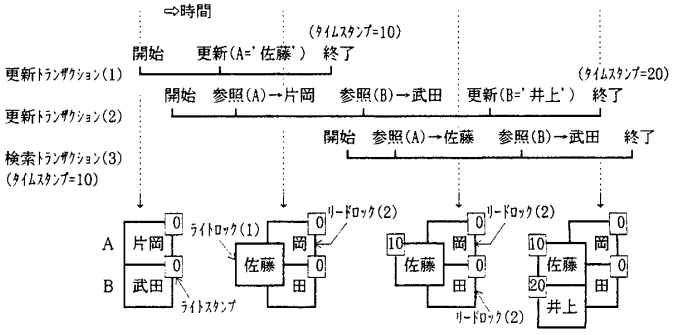

図３ ＭＶＨＣによる制御

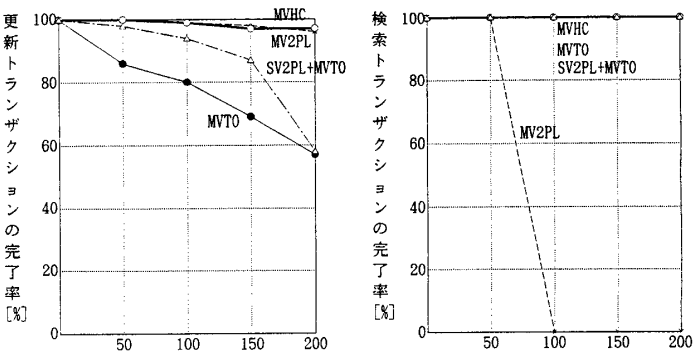

図４ 性能評価結果