アクセスに偏りがある分散データベースシステムにおける投機的トランザクション処理の性能評価

小澤 武志　P. Krishna Reddy　喜連川 優

東京大学生産技術研究所

〒 106-0032 東京都港区六本木 7-22-1
Phone: 03-3402-6231, Fax: 03-3479-1706

{kozawa,reddy,kitsure}@tkl.iis.u-tokyo.ac.jp

あらまし 本稿では，トランザクション処理の高性能化手法である投機的トランザクション処理手法に対する性能評価を行った．今回用いたシミュレーションモデルはアクセスに偏りのある分散データベースを想定している．アクセスに偏りのある場合には，トランザクション同士のデータ競合が起こりやすいため，通常の場合と比較してトランザクション処理性能は低下する．投機的トランザクション処理手法はこのような場合において従来の手法と比較してより優れた性能を発揮する手法であることをシミュレーションを通して証明する．

Performance evaluation of Speculative Transaction Processing on distributed database system with skew access

Takeshi Kozawa　P. Krishna Reddy　Masaru Kitsuregawa

Institute of Industrial Science, The University of Tokyo

Abstract In this paper, we evaluate performance of speculative transaction processing method on distributed database system with skew access. In this case, performance of transaction processing deteriorates, because data contention between transactions is becoming more significant. Through simulation experiments, we prove that speculative transaction processing method is superior to conventional method in such case.

1 はじめに

近年のデータベースシステムにおいては，インターネットの急速な普及により，ネットワークを通じてのアクセスが増加している．さらには，電子商取引 (Electronic Commerce:EC) が徐々に一般に普及しつつある．これらの影響により，データベースシステムにおけるトランザクション処理はその処理数の増加や分散化といった傾向を見せている．

また，計算機環境に関しては，プロセッサ処理速度は著しい向上を見せており，コストパフォーマンスも飛躍的に向上している．そして，主記憶は大容量・低価格化が非常に進んでいる．一方で，通信性能に関しては，その帯域は大きく進歩しているものの通信遅延は依然として大きい．

上記のような状況においては，従来の 2 相ロック (2 Phase Lock:2PL)[1, 2] や楽観的 (Optimistic) 手法 [3] のようなトランザクション処理手法を分散データベース環境に応用しただけでは，トランザクション間のデータ競合の増大や連鎖的なアボート等の現象が生じやすく，十分な性能を発揮することは難しい．

これに対して，我々は豊富なコンピュータリソースを活用してトランザクション処理を行うことで，トランザクションの処理性能をより高める手法として，投機的トランザクション実行手法 [4] を提案すると共に，シミュレーションによる評価を行い，この手法が従来の手法に対して有効であることを示した．

本報告では，この投機的トランザクション実行手法に対して，データ競合が多く発生するため，通常に比べてさらにトランザクション処理性能の低下を招く，データ毎のアクセスに偏りのある場合を想定したシミュレーションを行い，この結果をもとに従来の手法に対して大幅な性能向上が可能であることを示す．

2 従来のトランザクション処理手法

2.1 代表的なトランザクション処理手法

ここでは，従来から用いられている代表的なトランザクション処理手法を簡単に説明する．

2.1.1 2 相ロック手法

2 相ロック (2PL) 手法[1] [1, 2] では，データは使用時に必ずロックされる．ロックされたデータにアクセスしようとする他のトランザクションはロックが解放されるまで待機しなければならない (図 1)．

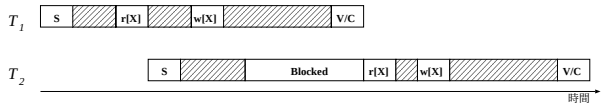

図 1: 2PL

2.1.2 楽観的手法

楽観的手法 [3] では，トランザクションは実行中にはデータに対してロックをしない．処理を終了した時に，その処理結果が直列化可能性を侵さない場合にはコミットされ，そうでない場合は，アボートされた後にリスタートされる．図 2 は，最も単純な楽観的手法のにおける処理例を示している．

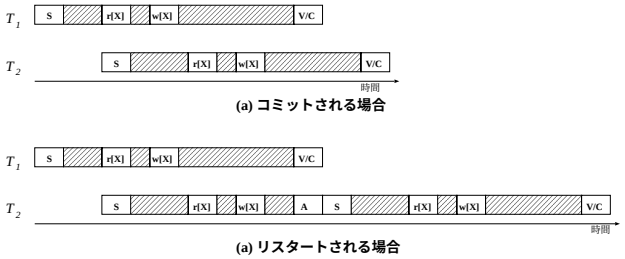

図 2: 楽観的手法

2.2 Speculative Concurrency Control

ここでは，集中データベース環境におけるトランザクション処理性能向上の研究例である，Speculative Concurrency Control (SCC)[6] について述べる．

[1] 悲観的手法とも呼ばれる．

SCCは実時間データベースシステムにおける並行制御法であり，データベースの整合性を脅かす衝突が発見されたらすぐに，代りのシャドウトランザクションがスタートし待機する．この方式を利用することにより，トランザクションのレスポンスを向上させることが可能となる．図3にSCCにおける処理の概略を示す．

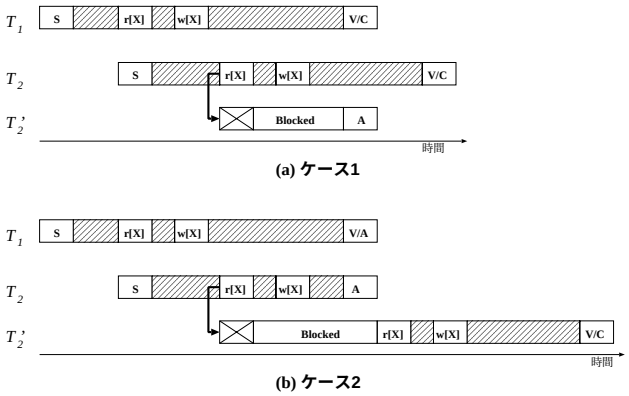

図 3: Speculative Concurrency Control

3　投機的トランザクション処理機構

投機的トランザクション処理手法の基本的な考え方は，トランザクションの処理に可能な限り多くのメモリやCPUなどのリソースを投入することで，その処理性能を向上しようというものである．この手法では，トランザクションがデータにアクセスする際に，オリジナルのコミット済みの値と，先行するトランザクションにより更新された(しかしコミット処理中の)値の両方に対して処理を行う．このコミット処理中の値に対する処理を投機的実行と呼ぶ．

投機的実行を行ったトランザクションはコミット処理時に，先行するトランザクションがコミットされたかアボートされたかによって適切な実行を選択する．つまり，トランザクション実行中に更新された値に関して投機的実行を行うことで，衝突しているトランザクション処理を直列化可能性を侵すことなく並列に行うことができ，トランザクション処理の高性能化が可能となる．

投機的手法のアルゴリズムの説明として，ロック両立性とデータ管理機構について以下にとりあげる．

3.1　ロック両立性

Lock requested by T_j	**Lock held by T_i**	
	R	**W**
R	○	×
W	×	×

(a)

Lock requested by T_j	**Lock held by T_i**		
	R	**EW**	**SW**
R	○	×	○*
EW	○*	×	○*

*T_jは投機的実行を行う

(b)

表 1: (a)2PL (b) 投機的手法 におけるロック両立性

表1において，○は対応するロック要求が両立できることを示し，×は対応するロック要求が両立できないことを示している．R，Wはそれぞれ Read，Write ロックを表している．2PLにおけるロック両立性は表1(a)に示されている．

投機的手法においては，W ロックは EW(Execution Write)，SW(Speculative Write) ロックに分けられる．トランザクションはRロックとEWロックのみを要求する．トランザクションがデータを更新すると，EWロックはSWロックに変わり，そのトランザクションはコミット処理に入る．投機的手法手法のロック両立性は表1(b)に示されている．

3.2　データ管理機構

投機的手法では，コミットされていない更新データに基づき投機的実行を行う．そこで，データが複数のバージョンを扱えるようにするため，各データは木構造で表現される．その構造はルートとしてコミットされているバージョンを持ち，残りの

ノードにコミットされていないバージョンを持ったツリーとなっている．以下にT_1, T_2, T_3の 3 つのトランザクションがデータ X, Y, Z, W に対して，

$$\begin{aligned} T_1 &: r_1[X]w_1[X]r_1[Y]w_1[Y] \\ T_2 &: r_2[X]w_2[X]r_2[Z]w_2[Z] \\ T_3 &: r_3[X]w_3[X]r_3[W]w_3[W] \end{aligned}$$

という処理を行う場合について，投機的手法におけるデータ管理手法の原理を説明する (図 4)．

最初は，それぞれのデータは，X_1, Y_1, Z_1, W_1 という既にコミットされている 1 つのバージョンのみで構成されている．まず，T_1が X, Y にアクセスすることにより処理を行い，X, Y はそれぞれ $(X_1(X_2)), (Y_1(Y_2))$ のように更新される．この後T_1はコミット処理に入る．次に，T_2は X, Z にロックを獲得するが，T_1がコミット処理中であるため，それぞれのデータに対して 2 つの投機的実行を行う．その結果，データ X, Z はそれぞれ $(X_1(X_2(X_4), X_3)), (Z_1(Z_2, Z_3))$ のように更新される．同じように T_3 は 4 つの投機的実行を行い，処理が終了したときには，データ X は $(X_1(X_2(X_4(X_8), X_6), (X_3(X_7)), X_5))$，$W$ は $(W_1(W_2, W_3, W_4, W_5))$ のように更新される．

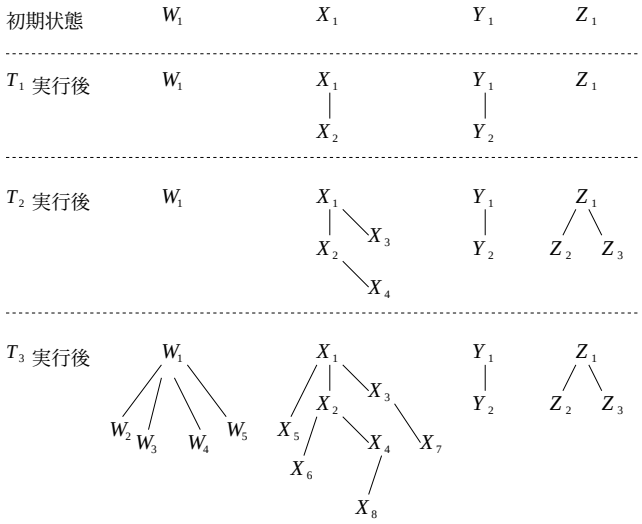

図 4: データ管理機構

この後，各トランザクションが終了した時点で，その結果に応じてデータのバージョンツリーが更新される．例えばデータ X はT_1がコミットされた場合には X_1，アボートされた場合には X_2を新たな X のルートとして更新される．

4 アクセスに偏りがある場合のトランザクション処理の性能評価

4.1 シミュレーションモデル

今回作成したシミュレータが想定しているシステムを模式的に簡単に示したものが図 5 である．システムは 4 つのデータベースサイトで構成されて，それぞれのサイトは他の全てのサイトとネットワークで接続されている．ネットワークを通じて他のサイトと通信を行う際の遅延時間は，どのサイト間でも同一である．また，それぞれのデータは 1 つのサイトのみに存在している．

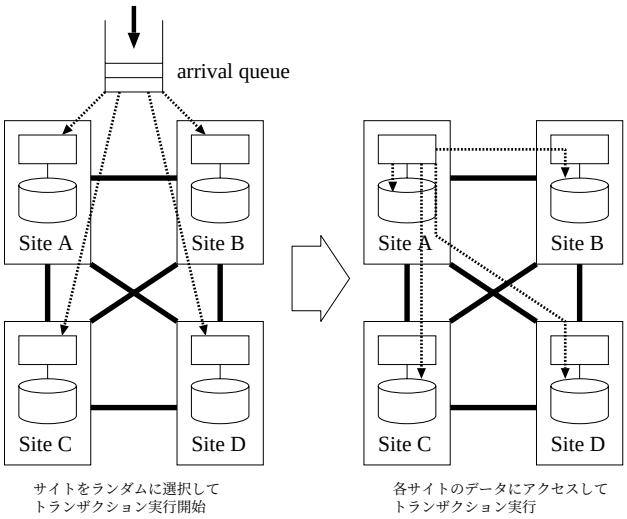

図 5: シミュレーションモデル

すべてのサイトはトランザクションマネージャとデータマネージャの両方の機能を持っている．トランザクションマネージャはトランザクションの処理を管理し，一方データマネージャは個々のデータベースを管理する．システム内にトランザクションが到着したら，そのトランザクションはランダムに決定されたホームサイトで処理を開始する．トランザクションは自分のサイトそして他のサイトに存在しているデータにアクセスして処理を実行する．ネットワークはデッドロック等が発生したら，その時点ですぐに他のサイトに伝達さ

れるとする．また，あるサイトが他のサイトにメッセージを送ったとき，ネットワークは常にメッセージを渡されたのと同じ順番で届けるものとする．

4.2 シミュレーション条件

今回は，表2のような条件のもとでシミュレーションを行った．この表において，同時実行トランザクション数はシステム内で同時に実行されるトランザクション数の最大値を表している．また，投機限界とは投機的手法において1つのトランザクションが行う投機的実行数の最大値であり，これを変化させることにより使用可能なリソースが限られた状況を実現している．

このシミュレーションでは，アクセスに偏りのあるデータベースを想定している．ここで導入されているアクセスの偏りの変数が，例えばx_yの場合には，トランザクションが各サイトのある特定のx%の領域内のデータにアクセスする確率がy%であるということを示している[2]．

パラメータ	値
データベースサイズ	100 ~ 20000 objects
サイト数	4 sites
トランザクションサイズ	2 ~ 10 objects
CPU 処理時間	15 msec.
I/O 処理時間	35 msec.
通信遅延	2000 msec.
ローカルアクセスの割合	0.60
同時実行トランザクション	50
投機限界 (投機的手法のみ)	2 ~ ∞
アクセスの偏り	50_50 ~ 15_85

表 2: シミュレーションパラメータ

今回のシミュレーションでは，投機的手法に関しては無限投機の場合と有限投機の場合についてそれぞれ性能測定を行っている．また，比較のため2PLに対しても同様に性能を測定している．

[2] つまり $x = y$ の場合には，サイト内の全データオブジェクトにおいて，トランザクションがアクセスする確率は一定であり，どのデータにアクセスするかはランダムに決定される．

4.3 アクセスの偏りと処理性能

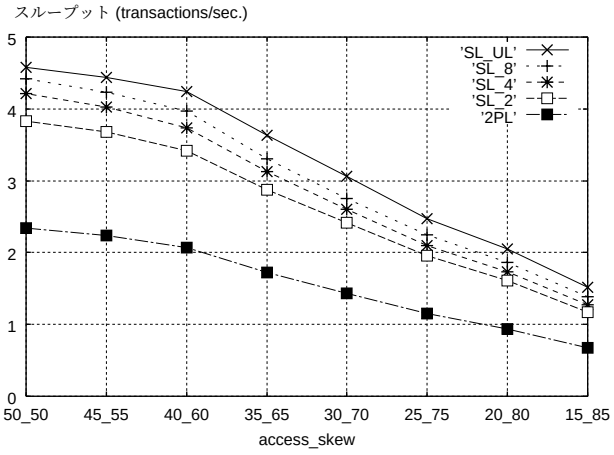

図 6: アクセスの偏りとスループット

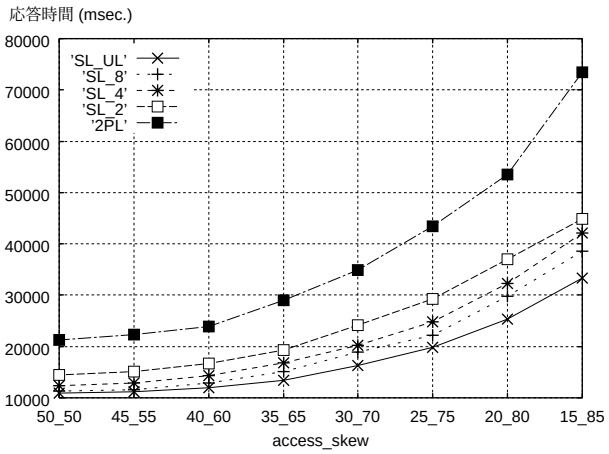

図 7: アクセスの偏りと応答時間

2PLと投機的手法 (投機限界2〜∞) について，アクセスの偏りの度合を変化させた場合のスループットと応答時間の変化を，それぞれ図6と図7に示す．そして，図8は，投機的手法のスループットの2PLに対する比を表したものである[3]．

図6，7より，アクセスの偏りが大きくなるにつれ性能は低下するものの，2PLと比較した場合には十分に高い性能を発揮していることが分かる．そして，図8より投機的手法はアクセスの偏りが

[3] 以降のグラフにおいて，SL_nは投機限界がnの投機的手法を表し，SL_ULは投機限界が∞の投機的手法を表す．

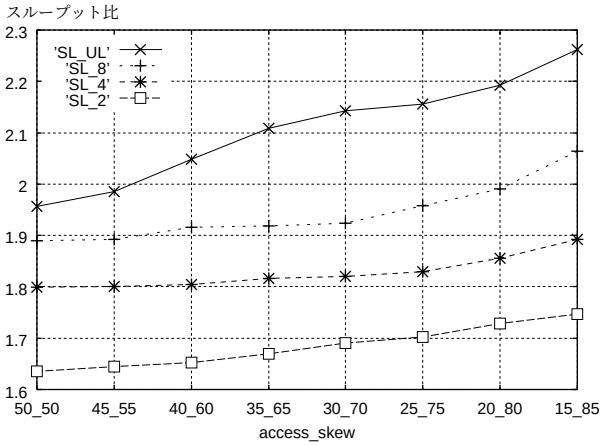

図 8: アクセスの偏りとスループット比

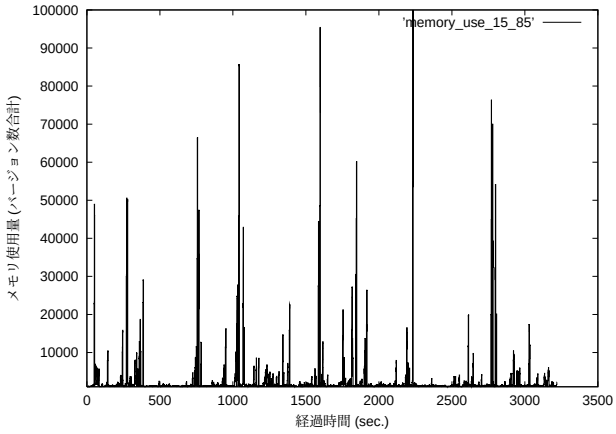

図 10: メモリ使用量の変化 (偏り 15_85)

大きくなればなるほど，2PL と比較して優位であることが分かる．

4.4 アクセスの偏りとメモリ使用量

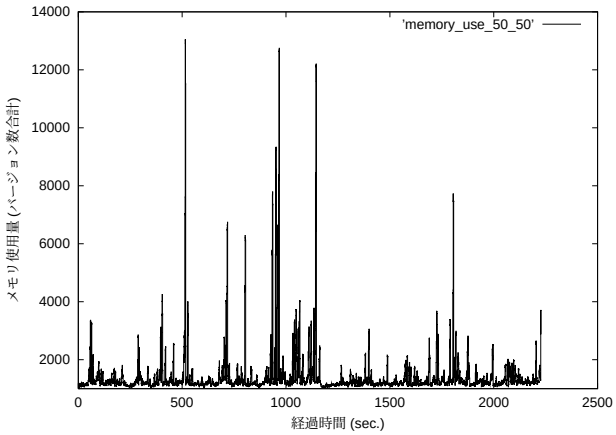

図 9: メモリ使用量の変化 (偏り 50_50)

投機的手法において，アクセスの偏り小さい場合 (偏り 50_50) と，大きい場合 (偏り 15_85) のメモリ使用量の変化を，それぞれ図 9 と図 10 に示す．ここでは投機限界は ∞ としている．

今回のシミュレーションでは，データのバージョンツリーの 1 ノードを記憶するために必要なメモリ量を，使用メモリ量の単位としている．したがって，ここでの使用メモリ量とはシステム内で発生している全てのバージョンの合計 ($\geq db\_size$) を表している．

この結果から，偏りが大きくなるにつれて　アクセスが集中するため，一般に性能の劣化が見られるが，投機的手法手法では，資源を更に投入することにより，偏りが大きくとも，通常の 2PL に比べて高い性能を維持することが可能であることが分かる．

4.5 データベースサイズが変化した場合

この節では，データベースサイズを変化させることで，トランザクションのデータ競合の度合を変化させ，それに伴う性能の変化を測定している．

図 11，12 はそれぞれデータベースサイズの変化に伴うスループットと応答時間の変化を表している．

データベースサイズが十分に大きい場合には，ほとんど競合が起こらない．データの競合がない場合には，投機的トランザクション処理手法は 2PL と全く同じ動作をするため，このような場合では投機的手法と 2PL はほぼ同等の性能となっている．

一方で，データベースサイズが小さくなるにつれて，データの競合の度合が増すため，2PL と投機的手法の両方とも徐々に性能は低下していくが，低下の割合は 2PL の方が大きいことが分かる．

図 13 は，投機的手法のスループットの 2PL に対する比を表したものである．この図から，投機的

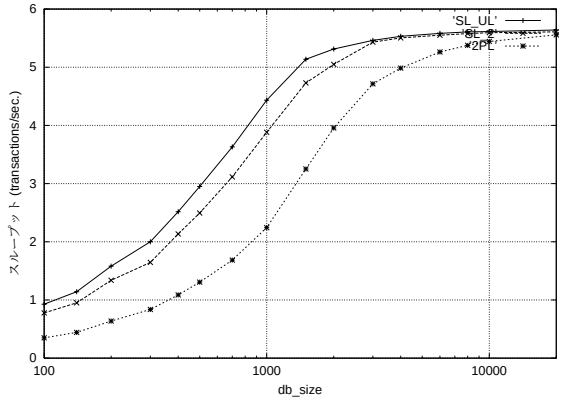

図 11: データベースサイズの変化とスループット

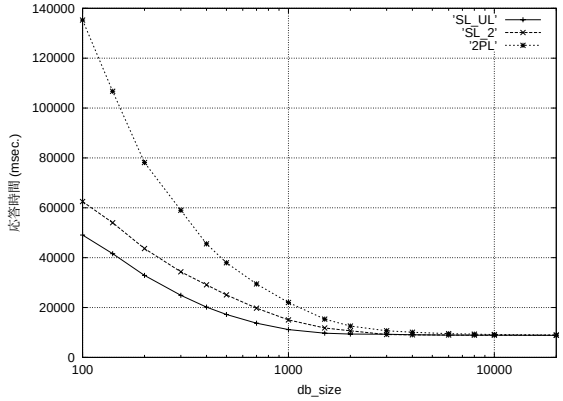

図 12: データベースサイズの変化と応答時間

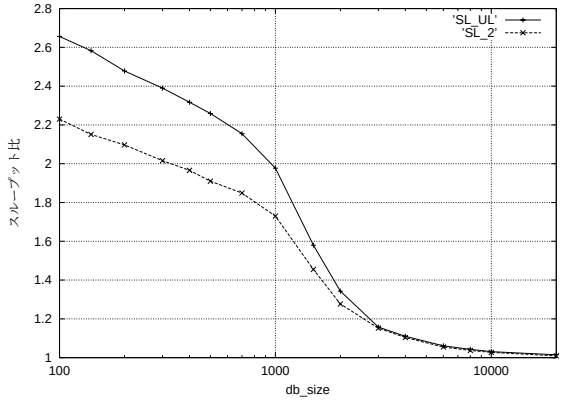

図 13: データベースサイズの変化とスループット比

手法はデータベースサイズが小さい，つまりデータの競合の度合が大きいような場合により性能を発揮する手法であることが分かる．また，投機限界が有限の場合でも，十分に高い性能を発揮できることが分かる．

5 まとめ

転送遅延の影響を受けやすい分散データベースにおいて，さらにトランザクションの性能低下を招くアクセスに偏りのある場合について，投機的トランザクション実行手法の性能をシミュレーションにより評価した．その結果，十分な資源を投入することで投機的手法は，このような状況においてより性能が発揮できることが証明された．また有限投機の場合のおいても 2PL に対して十分高い性能を発揮できることが分かった．

参考文献

[1] K. R. Eswaran, J. N. 5Gray, R. A. Lorie and I. L. Traiger, “The Notions of Consistency and Predicate Locks in a Data Base System”, *Communications of ACM*, 19(11), pp.624-633, 1976.

[2] J. N. Gray, “Notes on database operating systems: in operating systems an advanced course”, *Volume 60 of Lecture Notes in Computer Science*, pp.393-481, 1978.

[3] H. T. Kung and J. T. Robinson, “On optimistic methods for concurrency control”, *ACM Transactions on Database Systems*, 6(2), pp.213-226, June, 1981.

[4] P. K. Reddy and M. Kitsuregawa, “Improving Performance in Distributed Database Systems Using Speculative Transaction Processing”, *Proc. of the 2nd LASTED International Conference European Parallel and Distributed Systems*, pp.275-285, July, 1998.

[5] P. A. Franaszek, J. T. Robinson and A. Thomasian “Concurrency Control for High

Contention Environments", *ACM Trans. on Database Systems*, Vol.17, No.2, pp.304-345, 1992.

[6] A. Bestavros and S. Braoudakis, "Value-cognizant Speculative Concurrency Control", *Proc. of the 21st VLDB Conference*, pp.122-133, 1995.