モニタリングに基づいたグローバルデッドロック検出へのアプローチ

2Q-6

小寺誠、坂本明史、川上英、疋田定幸
沖電気工業(株)

1 はじめに

高性能で安価なワークステーションやミニコンピュータと高速のネットワークの発展が、分散データベースを数年の内に実用のものにするかもしれない。分散データベースでは、グローバルデッドロック(複数サイトにまたがるデッドロック)は回避できない問題である。複数のデータベースサイトにわたるアクセス処理(トランザクション)が検索処理に限られるならば、グローバルデッドロックは生じないように見えるかもしれない。しかし、サイトに閉じた更新処理を許せば、いずれグローバルデッドロックは現われる [Kam84]。システムの信頼性とデータの可用性のために、データは複数サイトで重複され、或いは分割して格納されるかもしれない。複数サイトにまたがる更新処理は必須となる。グローバルデッドロックは、分散データベースが直面しなければならない困難である。

グローバルデッドロックが頻繁に生じるか否かは、議論の多い問題であるが、対応する機構を省くことはできない。頻繁に生じないとしても、生じれば与える影響は大きい。システムがデッドロックを処理する機構を備えていなければ、影響は破滅的となる。頻度が高いのならば、検出のための機構は必須だろう。ただし、検出のために生じる負荷は低くなければならない。グローバルデッドロックの生じる頻度が高かろうと低かろうと、システムの処理効率を低下させるのは望ましいことではない。

我々は、homogeneous な分散データベース環境で、ロッキングとモニタリングを前提とした検出メカニズムに関する検討を行ってきた。同時実行制御方式として、ロッキング方式とタイムスタンプ方式がよく知られている。L. Lamportは分散環境での統一されたタイマを提案している [Lamport78]が、実現されたシステムの多くがロッキングを用いている [Menansce80] 。本稿では、モニタリングに基づいたグローバルデッドロック検出のために、各データベースサイト外部から可視である必要のあるブロッキング情報に関する基本的な検討を記述する。

なお、本研究は、通商産業省工業技術院の大型プロジェクト「電子計算機相互運用データベースシステムの研究開発」の一環として研究開発を進めているものである。

2 アプローチ

デッドロックの検出は待ち合わせグラフ(Wait For Graph)中のループの検出である。以下では、待ち合わせグラフをWFGと略称する。グローバルなWFGの効率的な生成の方式を論じる。複数のサイトにまたがってデッドロックが発生するとき、そのデッドロックはグローバルである。複数サイトにまたがる、トランザクション間の待ち合わせ関係はグローバルWFGが記述する。WFGが生成された後のデッドロック検出に関しては、すでに膨大な研究結果が報告されている [Knapp87]。以下で論じるのは、各サイトからグローバルWFGに属するノードとエッジを取り出すかである。言い換えれば、グローバルWFGを正しく生成するために、モニタできる(可視であることが必要な)ブロッキング情報に関する検討である。従って、このようなノードとエッジを発生させる可能性のある場合を中心に検討を行う。ノードとエッジの除去の方式と契機も検討が必要だろう。簡単のため、以下での検討対象となるロックモデルは、Single-Resource Model とする。

2.1 データベースのモデル

分散データベースはn個のサイトの集合{S1 ……Sn }から構成される。m個のトランザクションが存在し、t1 ……tm と表記される。各サイトのローカルなデータベース管理システム(LDBMS)は、サイト上で閉じたデッドロック(ローカルデッドロック)を検出する機能を持つと仮定する。

2.2 用語の定義

本稿で使う用語の意味を定義する。

Intersiteとintrasite blocking:

サイトSi 上のトランザクションta は、他のサイトSj のトランザクションtbに処理を中断(ブロック)させられるかもしれない。tb がロックしたリソースがtb の処理終了まではta によっては獲得されない。この状態を複数サイトにまたがったブロッキング(intersite blocking)と呼ぶことにする。一方、トランザクションta は、サイトSi 上のトランザクションtc がブロックするかもしれない。これはサイト内でのブロッキング(intrasite blocking)である。

Transactionの間のdependency:

あるトランザクションtm が他のトランザクションtn にブロックされた場合、tm はtn にdependent であると呼び、tm →tn と表記する。もし、二つのトランザクションが別のサイト上で開始されたものであれば、この関係はintersite dependent であり、そうでなければintrasite dependent となる。このようなdependency が列を構成することは可能である。たとえば、トランザクションt1がt2にブロックされ、t2がt3にブロックされている場合、この関係をt1→t2→t3と表記できるだろう。これをdependent sequenceと呼ぶことにする。もし、あるdependent sequenceに属するすべてのトランザクションが同一のサイトで開始されたものならば、そのdependent sequenceはlocal である。さもなければintersite dependent sequenceとなる。

Blocked transactionとblocking transactions:

Blocking transactionは、他のトランザクションをブロックしているトランザクションを意味する。ブロックされたトランザクションは、blocked transaction と呼ぶ。

3 WFGの生成と消滅

3.1 WFGの生成

グローバルWFGに属するエッジが発生する2つの場合に関して詳細に検討する。Si 上のトランザクションtp がSj 上のトランザクションtq にブロックされ、tp →tq のintersite blocking が生じたとする。

Case.1: tp はblocking transactionではない。または、tp を終端とするlocal dependent sequenceには、他サイトのトランザクションに対するblocking transactionが存在しない。

Case.2: tp はblocking transactionである。かつ、tp を終端とするlocal dependent sequenceには、他サイトのトランザクションに対するblocking transactionが存在する。

3.1.1 Case.1の詳細検討

Case.1に関しては、blocking transactionとblocked transaction の状態によって、以下の3つの場合に分けて検討を行う。

(1) tp →tq のintersite blockingが生じる前に、tq を終端とするintrasite dependencyが存在したとする。tp →tq のintersite blockingの発生時に、可視でなければならないブロッキング

A Monitoring Oriented Approach to Global Deadlock Detection

Makoto KOTERA, Akifumi SAKAMOTO, Suguru KAWAKAMI, and Sadayuki HIKITA

OKI Electric Industry Co., Ltd.

情報は、tp →tq で十分である。

(2)　逆に上記の例のようなlocal dependencyが存在しなかった場合を論じる。tp →tq のintersite dependencyの発生が先行し、tq を終端とするようなlocal dependent sequenceが順不同に生じたとする。最終的に、このlocal dependent sequenceの終端に位置するトランザクションをblocked transaction とする新しいintersite blockingが生じた時に、このblocking情報ts →tt ::Sk のみを知らされたモニタは効果的には機能しない。必要とされるintersite blocking情報は、tq →tt である。このblockingは、サイトS3上でのみ可視なlocal dependent sequenceが引き起こす。このlocal dependencyが外部から不可視であることは、目的とするintersite dependencyの発生を認識する契機を失わせる。

(3)　ts →tt のintersite dependencyがすでに存在する場合、tp →tq のintersite blockingにともなって可視である必要のあるのは、tp →ts または、tp →(tq →……→ts)のいずれかとなる。グローバルWFGに属するべきエッジは、tp →tq と tp →ts であり、tp →(tq ……t9)が可視でなければならないことを暗示している。一般的な規則は、Case.2の詳細な検討が導き出すだろう。

3.1.2　Case.2の詳細検討

ts →tt のintersite blocking の発生が、tp →tq のintersite blocking に先行した場合、可視であるべきブロッキング情報を論じる。この場合、intrsite blockingが重要なintersite blocking を引き起こすことが銘記されなければならない。ポイントは、ts →tt が十分であるか否かである。以下の2つの場合に分けて考察を行う。

(1)　tq よりts に至るlocal dependencyがすでに生じているとすると、ts →tt が可視であれば十分である。tp →tq が生じると、グローバルWFGに既存なtq →ts のエッジが、tp →tq →ts →tt に至るintersite dependencyの生成をモニタにただちに認識させる。

(2)　上記のlocal dependencyが未生成の場合、問題は、Case.2の一般形式となる。グローバルWFGには、ts →tt とtp →tq のエッジが存在する。tq とts を、それぞれ始端と終端とするlocal dependent sequenceの発生は、グローバルWFGに新しいエッジ、tq →ts を付け加えることを必要にする。以上のような観察は、Case.1での議論を含めた一般規則が、以下のように表わされることを示す。

3.1.3　一般的な規則

一般規則Ⅰ：　Eventual blocked transactionとeventual blocking transactionが共に存在すれば、これらのトランザクションは可視である必要がある。Blocked transactionの属するlocal dependent sequentの始端側に位置するトランザクションで他サイト上のトランザクションに対するblocking transactionとなっているものが、eventual blocked transactionとなる。一方、blocking transactionの属するlocal dependent sequence内の終端側にあり、他サイト上のトランザクションによるblocked transaction となっているものをeventual blocking transactionと呼ぶ。

3.2　WFGの収縮

グローバルWFGの収縮(shrinking)の方式と契機を論じる。他サイトのトランザクションをブロックしているトランザクションの終了が、グローバルWFGからエッジを取り除く契機となる。最小限、blocking transactionの終了が可視でなければならない。

正常あるいは異常にかかわらず、intersite dependent sequenceの終端に位置するトランザクションの終了に注意を集中するのは正しい選択に思える。障害を考慮しなければ、blocked transactionがデッドロックの解決の副作用として以外にはabort されないと仮定するのは妥当である。Blocked transaction がblocking transactionの終了前に正常終了するデータベースシステムは、データのconsistency を効果的に保証しないだろう。厳格な2相ロックのshrinking phase は、トランザクションの終了と同時に行われる [Gray79] 。さもなければ、データベースはinconsistentとなる。検討する必要があるのは、以下の2つの場合に限られる。

Case.1　Intersite dependent sequence の終端に位置するトランザクションが正常に終了した。

Case.2　Intersite dependent sequence の終端に位置するトランザクションを含んだデッドロックが解決され、そのトランザクションがabortされた。

3.2.1　Case.1の詳細検討

トランザクションtt の終了に伴なうts →tt の消滅は、tsをblocking transactionとするlocal dependencyを示すエッジのグローバルWFGからの除去を促さなければならない。たとえば、ts →tt が消えれば、tq →ts のエッジは存在理由を失う。3.1.2 の(2) での詳細な検討の結果も、この結論を支持する。ts の、他のトランザクションによるintersite blockingの発生に際しても、正しいグローバルWFGの生成は保証される。だとすれば、規則は以下のように一般化できる。

一般規則Ⅱ：　Blocking transactionの終了は、blocked transaction を終端とするlocal dependent sequence のelementで構成されるエッジのグローバルWFGからの除去を意味しなければならない。

3.2.2　Case.2の詳細検討

Abort されたトランザクションのための特別の検討が不要であることは自明である。Abort されたトランザクションが、他のサイト上のトランザクションに対してblocking transactionとなっていないならば、これはCase.1である。Blocking transactionは正常に処理を終了できる。従って検討は、blocking transactionがabort される場合に集中される。処理の終了が正常であるか異常であるかにかかわらず、処理を終了したトランザクションがロックしたリソースは解放され、blocked transaction に通知されるだろう。

4　要約と今後の拡張

本稿では、モニタリングに基づいたグローバルデッドロック検出のための基本的な検討を記述した。最初に分散データベースの簡単なモデルと用語を定義し、Single-Resource Model のもとで、グローバルなWFGを正しく生成・収縮させるためにモニタできなければならない（可視でなければならない）ブロッキングの情報について論じた。

本稿での検討は最も基本的な環境を前提にして行われた。本稿の検討結果の、AND-Model を前提とした拡張はただちに行われなければならない。分散環境で必ず生じる各種障害に関する議論、分散データベース管理システムとローカルデータベース管理システムとの機能分担も、分散データベースの実現には不可避の検討項目となるだろう。

参考文献

[Gray79]　Gray, J. N., Notes on Database Operating Systems., Operating Systems: An Advanced Course, Bayer, R., Graham, R. and Seegmuller, G. (editors), Springer-Verlag, 1979

[Kam84]　Kambayashi, Y. and Kondoh, S., Global Concurrency Control Mechanisms for a Local Network Consisting of Systems without Concurrency Control Capability., In Proc. of National Computer Conference, 1984

[Knapp87]　Knapp, E., Deadlock Detection in Distributed Database, ACM Computing Surveys., Vol. 19, No. 4, December 1987

[Lamport78]　Lamport, L., Time, Clocks, and the Ordering of Events in a Distributed System., CACM, Vol. 21, No. 7, July 1978

[Menansce80]　Menansce, D. A., et. al., A Locking Protocol for Resource Coordination in Distributed Databases, ACM TODS, Vol. 5, No. 2, 1980