データベース・システム 68－4
（1988. 11. 17）

時計サイトと先読みスケジューラを利用した分散データベース並行処理方式

李 吉桂　　茨木 俊秀

京都大学工学部

あらまし 文献[5]によって先読みスケジューラによる分散データベースシステムの並行処理制御方式が提案されたが，この方式では時刻印(time stamp)の決定も分散アルゴリズムによってなされている．これに対し，本報告では，時刻印を供給する共通時計サイト(central clock site) の存在を仮定し，新しい並行処理方式を提案する．この方式は前方式にくらべ，通信量が少なくてすみ，また簡単である．さらに主な故障に対し，それらへの対処方法についても検討する．

Distributed Database Concurrency Control by Cautious Schedulers with Central Clock Site

Jigui LI　　Toshihide IBARAKI

Faculty of Engineering, Kyoto universty

Abstract The distributed concurrency control algorithm proposed in [5] for a distributed database system that uses a cautious scheduler at each site is based on the time stamps issued by a distributed algorithm. In this paper, we propose a modification of the algorithm in that the time stamps are issued from the central clock residing at a certain site in the system. This algorithm is much simpler and requires less number of messages than the previous algorithm. It is also discussed how system failures are treated to maintain basic functions of a database system.

1．まえがき

分散データベースシステム(DDBMS, distributed database management system)[3]の並行処理のために，従来の２相施錠方式や時刻印方式[1]による方法にかわり，先読みスケジューラ[6]による並行処理制御の方法が[5]によって提案されている．本報告ではさらに通信コストを低減し，制御手順を簡単にするために，中央時計サイトを利用した先読みスケジューラによる並行処理方式を提案する．便宜上,[5]に提案された制御方式を**分散時計方式**と呼び，本報告の制御方式を**共通時計方式**と呼ぶ．共通時計方式による並行処理は比較的簡単であり，しかも，デッドロック (deadlock)を生じることはなく，後退復帰も必要ないという先読みスケジューラの特色をそのまま継承しているので，実用上意味があると考えられる．本報告ではさらに主な故障に対し，その対処の仕方を検討する．

2．データベースモデルと先読みスケジューラ

2.1 データベースモデル

データベースシステムはデータ項目の集合D={X, Y, …}とトランザクションの集合T={T0, T1, …, Tf}からなる． 一つのトランザクションTiはいくつかの読取りステップRi[S]と書込みステップWi[S]の系列である． ただし，S,S´⊂D．各x∈Sに対する,Ri[X]はデータ項目Xの内容を読み取るという操作であって，X∈S´に対するWi[X]はデータ項目Xの内容を更新することを意味する．T0=W0[D]は他のトランザクションの実行に先立ち,全てのデータ項目の初期値を準備し,Tf=Rf[D]は全てのトランザクションの後で，全てのデータ項目の最終値を読み取る．なお，各トランザクションは，現在要求中のステップが完了してはじめて次のステップを出すことができると仮定している．

W0[D]を最初に，Rf[D]を最後にもち，その間にすべてのTiのステップが並んだ系列をスケジュール(schedule)という．一つのスケジュールにおいて，それぞれのトランザクションの全ステップが他のトランザクションのステップと混合することなく，ひとかたまりとなって実行されているならば,直列スケジュール(serial schedule)であるという.スケジュールsがある直列スケジュールs´と等価ならば**直列可能**(serializable)であるという[1,2].データベースの一貫性を保証するために，スケジューラは直列可能スケジュールを出力することが要求される．直列可能性の判断のために,スケジュールのTIOグラフ(transaction input-output graph)[6]を次のように定義し，TIO(s)と記す．TIO(s)は，各Ti∈Tに対応する節点を持ち，TjがXをTiから読み取る時，ラベルX付きの有向枝（Ti,Tj）を持つ．さらに，ある書込み操作Wi[x]に対し，どのTjもXをTiから読み取らない時，Wi[X]は不用(useless)であるといい，新しい節点Ti´を追加し，ラベルXを持つ有向枝（Ti, Ti´）を付す．このようにして得られるラベル付き有向グラフがTIO(s)である．

つぎに，TIO(s)の全節点を左から右に一列に並べる操作を考える，すなわち，TIO(s)の節点集合に全順序を付し,節点Tiが節点Tjの左にあることをTi《Tjと記す．この全順序《がつぎの条件を満たす時,DITS(disjoint interval topological sort)であるという．

(a)Ti《TjならばTIO(s)において,TjからTiへの有向路は存在しない．

(b)排除規則：h≠jを満たし，同じラベルを持つ有向枝（Th,Ti）と（Tj, Tk）に対し,Th《TkならばTi《Tjが成立する．

定理１[6]．スケジュールsが直列可能であるための必要十分条件は，TIO(s)にT0を最も左,Tfを最も右に置くDITSが存在することである．■

排除規則を陽に示すため，(b)のThからTkへ路が存在するとき，排除枝(exclusion arc)（Ti,Tj）を付す．可能な排除枝をすべて加えた結果をTIO(s)の閉包(closure)と呼び，TIO*(s)と記す．

2.2. 先読みスケジューラ(cautious scheduler)

各トランザクションTiはその最初のステップを発するとき，Tiのステップの集合STEP（Ti）を予め宣言しておく必要がある．スケジューラのある時点を考え，Pをそれまでに出力された部分スケジュール，qをその時点で実行の可否を問われているステップ，さらに

DEL：スケジューラによってその実行を遅延されているステップの集合．

PEND：∪STEP（Ti）－｛Pq中のステップ｝

とする．一般に，DEL⊂PENDである．各トランザクションはその前のステップの実行が完了した時，始めて次のステップを発するから，DELには各トランザクションについて，たかだか１個のステップが含まれている．またWO[D]は常にPの先頭に位置し，Rf[D]は出力系列の最後に加えられる．

先読みスケジューラＣＳ（Ｃ）の手順

CS0（初期化）：P:=WO[D]，q:=（最初に到着したトランザクションTiの最初のステップ），DEL:=Λ（空きリスト），　PEND:=STEP(Ti)（Tiが宣言した操作の集合）．

CS1（完成テスト）：PEND:=PEND－｛q｝とし，Pqに対する完成テストを加える．テストが失敗すればCS2へ，成功すればqの実行を許可し，CS3へ進む．

CS2（qの実行の延期）：PEND:=PEND∪｛q｝とし，qの性質に応じて以下の一つを実行する．

(a)q∈DELかつDELの全てのステップの完成テストが失敗した場合：CS5へ進む．

(b)q∈DELであるが(a)ではない：リストDELの次のステップをqとみなしCS1へ戻る．

(C)q∉DEL：qをDELの最後尾に加え，CS5へ進む．

CS3（qの実行）：P:=Pqとする．q∈DELならDELからqを除く．

CS4（DELのテスト）：DEL≠ΛならばDELの次ステップ（すなわち，まだテストされていない先頭のステップ）をqとみなし，CS1へ戻る．DEL=ΛならばCS5へ進む．

CS5（次の到着ステップ）：qを次に到着するステップとする．もしqがあるトランザクションTjの最初のステップならば，TjをＣＳ（Ｃ）に登録し，PEND:=PEND∪STEP（Tj）と置く．CS1へ戻る．■

なお，CS1の完成テストは，直列可能スケジュールのクラスSRの適当な部分クラスＣに対して行われ，PEND内のステップを適当に並んで得られる系列ＱでＰｑＱＲｆ［Ｄ］∈Ｃをみたすものが存在すれば成功し，そのようなQが存在しなければ失敗する．この目的に用いられるスケジュールのクラスＣには，SR，WW，WRWなどが考えられる（[6]等参照のこと）．

定理２[6]．先読みスケジューラＣＳ（Ｃ）はデッドロックに陥ることなく，最終的にＣに属すスケジュールを出力する．■

ところで，先読みスケジューラを実際に構成するには完成テストの実行方法を与えなければならない．ここではクラスＣ＝ＷＷの場合のみを扱う（CS(WW)と記す）．完成テストに利用される活性TIOグラフ(active TIO graph)を以下のように定義し，ATIO（Pq,PEND）と記す．このグラフは，ＴｏとＴｆおよびすでにＰｑあるいはPENDにその最初のステップを送っているすべてのトランザクションTiを節点とし，Ｐｑ部分において，TjがＸをTiから読み取っているならば，ラベルＸの有向枝（Ti,Tj）を持つ，Ｐｑ部分の不用なWi[X]は追加節点Ti′へのラベルＸの有向（Ti，Ti′）で示す．次にPEND内のWi[X]やRi[X]はそれぞれラベルＸの有向枝（Ti,Ti′）および（Ti″,Ti）で示される．Ti′,Ti″はいずれも追 加節 点である．さらにクラスＷＷでは，各

X∈Dに対し，つぎのような枝を追加する．

(a)PqにおいてWi[X]がWj[X]に先行するとき，ww－枝（Ti,Tj）を付す，

(b)Wk[X]をPqにおけるXへの最後の書込み操作とする．このとき，すべてのWj[X]∈PEND に 対しww－枝（Tk,Tj）を，また，すべてのRj[X]∈PENDに対しwr－枝（Tk,Tj）を付す．得られたグラフをATIOww（Pq,PEND）と書く．

排除枝:同じラベルXを持つ（Th,Ti）と（Tj,Tk）において，h≠jかつThからTkへの路が存在するならば，枝（Ti,Tj）を付す．

可能な排除枝をすべて加えた結果を排除閉包(exclusion closure）といい，ATIOww*（Pq,PEND）と書く．

定理3[4]．与えられるPqとPENDに 対し，CS（WW）の完成テストが成功するための必要十分条件はATIOww*（Pq,PEND）が有向閉路を持たないことである．■

したがって，クラスWWに対する完成テストは多項式時間で可能である．

3．分散データベースシステム

分散データベースシステム(DDBS, distributed database system)は図１に示すように通信回路によって接続されたｎ個のデータベースシステム（サイト(site)と呼ぶ）からなる．モデルの詳細はほとんど[5]と同じであるので，次の基本的な仮定のみ述べて置く．

(1)各サイトは他のサイトにどのようなデータ項目があるかを知っている．

(2)各サイトはそれぞれトランザクション制御装置および先読みスケジューラCS（C）によって独立に制御されている．

(3)一つのサイトａから他のサイトｂへ発せられたメッセージは，有限時間後，ａから出た順序でｂに届く．

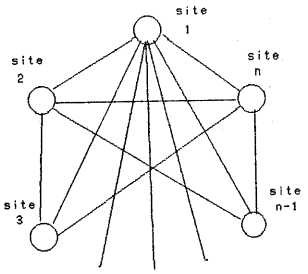
(4) 利用者がトランザクションを入力するサイトをそのトランザクションの根(root)サイトと呼ぶ．ある一つのサイト（根サイトと一致するとは限らない）ですべて処理できるトランザクションを局所(local)トランザクションと呼び，局所トランザクション以外のトランザクションを全域(global)トランザクションと呼ぶ．全域トランザクションにおいて根サイトで処理できるステップを局所ステップと呼び，局所ステップ以外のステップを遠方(remote)ステップと呼ぶ．

4．共通時計方式による並行処理

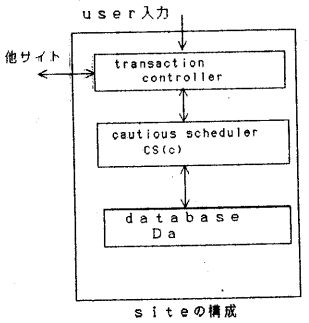
共通時計方式による並行処理は次のように実行される．

(1)ｎ個のサイトの中で，ただ一つのサイトは共通時計(central clock)を持っている．このサイトを時計サイトと呼ぶ（次に述べる時刻印の発行機能を除いて，普通サイトとまったく同じである）．

(2)Tgをユーザが根サイトへ入力した全域トランザクションとする．Tgはまず時計サイトへ必要な情報（Tgの名前，根サイト名，どのサイトのどのデータ項目に読み書きする予定であるか）を送る．ただし，読取りステップに対しては，そのデータ項目を持つサイトの一つを選び，書込みステップでは，そのデータ項目を持つすべてのサイトを選ぶ．このメッセージを受けた時計サイトはその時刻ｔgをTgの時刻印として決め，時計サイトはその時刻印と必要な 読み書 き情 報を 関連するサイト（Tg のステップ が実行さ れるサイトおよび 根サイト）へ伝送し，そのサイトの先 読みスケジューラに登 録する（図２）．時 刻 印が時 計サイトから 根サイト返送されたあと，根サイトはTgからステップ 実 行要 求 が出れば，それを直接にそのステップが実 行されるサイトへ 送る（そのステップは 必ずそこで



DDBMS



site の構成

図 1

実行される）．

(3) Thがユーザの入力した局所トランザクションであれば，時刻印を付す必要はなく，直接にそのトランザクションを処理するサイトへ送って（根サイトで実行される局所トランザクションであれば，送る必要はない），そのサイトの先読みスケジューラに登録する．

(4)先読みスケジューラにおける登録

(a)全域トランザクションTgの登録はつぎのように行う．根サイトでは，時計サイトからの時刻印をうけた後，Tgの根サイトでの局所ステップを時刻印とともに根サイトの先読みスケジューラに登録する．Tgからあるサイトｂへの遠方ステップについては，時計サイトが時刻印とともに，そこでの読取り集合と書込み集合をサイトｂへ登録する．各サイトｂでのTgの先読みスケジューラへの登録はつぎのように行なう．すなわち，Tgのサイトｂにおける読取りおよび書込みステップをサイトｂの先読みスケジューラのATIOグラフに，時刻印の制約枝とともに加える．時刻印制約枝とはATIOグラフ内のTiとTjが時刻印を持ち，Tiの時刻印がTjの時刻印より小さい時付される枝（Ti,Tj）のことである．

(b)局所トランザクションThの先読みスケジューラへの登録は，時刻印を付す必要はなく，集中データベースの場合と全く同じように行えばよい．

(5)ステップの実行

(a)全域トランザクションTgの根サイトでは，Tgの登録がそのサイトで済んでおりさえすれば（つまり，時計サイトからの時刻印を受けていれば），Tgの要求にしたがって，ステップの実行要求を関係サイトへ送る．局所トランザクションThについては，Thの要求をそのまま関係サイトへ送る．

(b)根サイトからトランザクションTのあるステップの実行要求を受けたサイトｂでは，Tの登録がすでに済んでおれば（すなわち，全域トランザクションの場合は，時計サイトからの時刻印と書込み及び読取り集合を受けていれば；局所トランザクションの場合はただちに），サイトｂの先読みスケジューラへそのステップの要求をだす．登録が済んでいなければ，登録が完了するまで待たせる．

(c)ステップの実行が完了すると，得られた結果（読取りステップの場合）とともに，完了したことを根サイトに知らせる．

5．共通時計方式の正当性について

上に提案した共通時計方式の正当性を検討する．

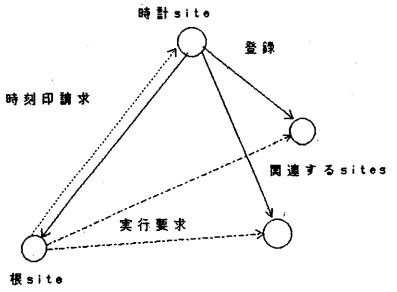


図２

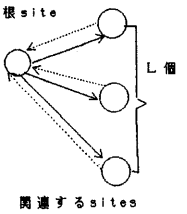


図３ａ

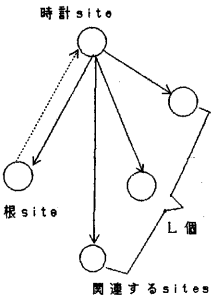


図３ｂ

まず次の定理を与える．

定理４． 各サイトｂに要求されたトランザクションの登録は拒否されることはない．

証明：時刻印のない局所トランザクションは先読みスケジューラの性質によって，必ず登録できる．全域トランザクションについては，時計サイトから一つのサイトｂへ時刻印の伝送ルートに沿って登録がなされるので，３節の性質(3)によって，登録要求は時刻印の順序でサイトｂに到着する．その結果，サイトｂの先読みスケジューラへ登録要求を出したトランザクションＴｇの時刻印はその時点のATIOグラフで最大であるから，Ｔｇから出る時刻印制約枝はない．したがって，Ｔｇが原因となる閉路がATIOグラフに生じることはなく，Ｔｇの登録が拒否されることはない．■

定理５． 共通時計方式によってすべてのトランザクションの処理が終了した時，各サイトにおいて，得られているTIOグラフを次のように重ね合わせて，全体のTIOグラフを作る．すなわち，同じ全域トランザクションを表す節点は一つに重ね合わせ，また全域トランザクションの間の時刻印制約枝は異なるサイトの間であっても加える．このようにして得られるTIOグラフはＴ０を最初，Ｔｆを最後にするDITSを持つ．

証明：文献[3]と同様に示せるので，省略する．■

この結果と定理１によって，この分散データベースシステムが出力したスケジュールは直列可能である．

定理６． 各トランザクションから出されるステップの処理要求はすべて有限時間内で実行できる．

証明：トランザクションが局所トランザクションであれば，先読みスケジューラの性質によって，ステップの処理要求はすべて有限時間内で実行できる．全域トランザクションは，まず定理４によって，有限時間内で登録できる．登録されたトランザクションのステップは予め宣言してある．したがって，時刻印制約枝がなければ，先読みスケジューラの性質によって，必ず有限時間で実行できる．そこで時刻印制約枝の影響を考える．定理５と同じやり方で，全てのサイトのATIOグラフをまとめて，全体のATIO（Pq,PEND）を作る．定理５と同様に，この ATIOグラフには閉路がなく，DITSを持つ．このようなDITSを一つ選び，節点をDITSの順に並べる．そこで，PEND（未実行）に属すステップの中で，時刻印が最小であるステップｒを選ぶ（複数個ステップがある場合にはDITSの中で一番左にあるトランザクションのステップで，次に実行要求の出されるものを選ぶ）．すなわち，ステップｒをPENDの中の最初のステップと考える．そのステップｒを実行するサイトをサイトａと置く．あきらかにサイトａのＣＳａ（Ｃ）内のATIOグラフはDITSを持つ．したがって，ＣＳａ（Ｃ）ではPENDの中のステップをDITSの順に並べたスケジ

ュールをｓとするとＰｑｓＲｆ[Ｄ]∈Ｃ，つまりｓの順に実行でき，しかもｒはｓの先頭に位置している[6]．このため，サイトａにおいてステップｒの実行は可能であり，ｒの実行要求が出されると，有限時間で実行される．この議論を繰り返せば，ステップの処理要求はすべて有限時間内で実行できることを示せる．■

定理７．共通時計方式ではシステム全体のデッドロックは起きない．

証明：定理６により，各サイトはそのサイトのステップをすべて有限時間内で実行できる．よって，システム全体のデッドロックは起きない．■

６．共通時計方式のメッセージ数

共通時計方式の主な特徴として，全域トランザクションの登録のためのメッセージ数は分散時計方式より少ない．全域トランザクションＴｇが根サイトを除いて，L個のサイトと関連するとしよう．分散時計方式ではＴｇの登録のため，図３ａのようにすくなくとも２L回のサイト間のメッセージ伝送が必要である．一方，共通時計方式でせいぜいL+2回のサイト間のメッセージ伝送でよい（図３ｂ）．

伝送される各メッセージの情報量は両方とも大体同じである．ただし，共通時計方式では，時計サイトへの通信量が他のサイトに比べて大きくなるので，サイトの通信能力を高くとるなどの考察が大切である．なお，あらかじめ，トランザクション取扱いの多いサイトを時計サイトとするなどの工夫を加えると，共通時計方式の通信コストをさらに低減することが可能である．

７．故障の対処について

共通時計方式における故障のうち，重要なつぎの２種について考察する．

(1)サイトの故障

(a)通常サイトの故障

故障サイトをｂとする，サイトｂにあるデータ項目を読み取ったり書き込んだりするトランザクションはその事実を知ると仮定し，そのようなトランザクションの集合をＭと記す．データベースの一貫性を守るため，Ｍの中でまだ終結（commit）していないトランザクションはすべて取り消さなくてはならない（取り消す手順は[7]参照）．つまり，各サイトはサイトｂの故障についてのメッセージを受け取ったあと，トランザクション集合Ｍの登録と取り扱いを中止し，Ｍの中でまだ終結されていないトランザクションを取り消す．この作業が各サイトで完了した後，サイトｂを除いて，各サイトはふたたび正常に動く．

(b)時計サイトの故障

時計サイトが故障すると，共通時計方式の欠点として，システム全体の機能は停止する．しかし，次に述べるように他のサイトを時計サイトに選び，故障したサイトと関係あるトランザクションを除くと，システムはまた正常に動く．

まず，ある手順によって，サイトｅを時計サイトに選び（一般には，リーダ選出の分散アルゴリスムを用いる），サイトｅに時刻印の発行能力を持たせる．登録済みのトランザクションは一般にはそのままにするが，故障サイトに関係あるトランザクションはすべて登録を取り消し，後退複帰する．登録のまま残っているトランザクションの中で最大時刻印を見つけ，t_{max} と置く．その後，新しい時計サイトの開始時刻を t_{max} より大きくとり，動作を開始すれば，以上に述べた共通時計方式はふたたび正常に動く．なぜならば，その後登録するトランザクションはその前の時刻印より大きい時刻印を持っているので，各サイトの先読みスケジューラの実行に影響はないからである．

(2)メッセージの消滅（リンクの故障）

(a)根サイトから時計サイトへの時刻印請求メッセージが消滅する.

この時，時計サイトは根サイトからの時刻印請求メッセージを受けず，しかも根サイトも時計サイトからの返答を受けない．この場合に，故障対処として，根サイトは一定時間後(timeout)，もう一度時刻印請求メッセージを出さなければならない.

(b)時計サイトから根サイトへ返送される時刻印メッセージが消滅する.

上の(a)と同様，根サイトはもう一度時刻印請求メッセージを出さなければならない.

(c)時計サイトから関連するサイトｂへの登録メッセージが消滅する.

この時，共通時計方式の(4a)と(5b)によってサイトｂでの登録はなされないので，Ｔgのステップの実行要求はいつまでも待たされる．故障の対処法として，ステップの実行要求を出してから，一定時間遅延のあと,まだ実行できなければ，Ｔgの登録を取り消し，最初から登録手続きをやり直さねばならない.

（ｄ)ステップの実行要求メッセージが消滅する.

この時，関係するサイトでの実行は行われず，したがって，Ｔgの終結もできない．やはり，故障の対処法として，一定時間遅延の後,Ｔgの登録を取り消し，やり直すことになる．しかしこのように一定時間の遅延にもとづいて決めるためには，各側面の代価が必要である.

謝辞　いろいろ御討論頂いた研究室の諸氏に深謝いたします．なお，本研究は一部文部省科学研究費によっている.

文　献

[1] P.A. Bernstein and G.Goodman, Concurrency control in distributed database systems, ACM Computing Survtys, 13, pp.185-221(1981).

[2] K.P. Eswaran, J. N. Gray, R. A. Lori,and I. L. Traiger, The notions of consistency and predicate locks in a database system, Commun. ACM, vol. 19, no.11, pp.624-633, Nov.1976.

[3] M.A. Casanova and P.A. Bernstein, General purpose schedulers for database systems, Acta Informatica, 14, pp.195-220(1980).

[4] S.Ceri and G. Pelagatti, Distributed Databases Principles & Systems, McGraw-Hill International Student Editions(1985).

[5] 原嶋秀次，茨木俊秀，先読みスケジューラによる分散型データベースシステムの並行処理制御，電子情報通信学会論文誌，J70-D,6,pp.1140-1148(1987).

[6] T.Ibaraki,T.Kameda and N.Katoh:Cautious Transaction Schedulers for Database Concurrency Control,IEEE Trans on Software Engineering, SE-14, pp.997-1109(1988); 茨木他．Cautious scheduler（先読みスケジューラ）による並行処理制御，情報処理学会データベースシステム研究会,48-3,1985.

[7] 茨木俊秀,亀田恒彦，不完全な宣言の下での先読みスケジューラ，電子情報通信学会，データ工学研究会，DE87-13，1987年10月.