データベースシステム 107－1
（1996. 3. 11）

バルクアクセス・トランザクションを用いたデータウェアハウスの構築

大森 匡

電気通信大学 大学院 情報システム学研究科
情報システム設計学専攻 データベース学講座

内容梗概： 本稿では自律分散型組織に適したデータウェアハウスの構築モデルとして、「自律分散メンバ全体に関する大域的な情報を表すデータベースをデータセンタに用意し、その最適性を 参加メンバごとに自律分散的に維持する」、という枠組を提案する。さらに、この枠組を メンバごとに **BAT** (バルクアクセス・トランザクション、大量データをアクセスするトランザクションのこと) を実行することで実現する方法を述べる。 本稿では、具体例として、組織全体に関する大域的推定モデルや大域的現在状態を表すデータベースの最適性を自律分散的にログデータを使って管理する事例をあげ、提案方式の有効性を示す。

A Design Method of Data Warehouses Using Bulk-Access Transactions

OHMORI,Tadashi

The University Of Electro-Communications
Graduate School Of Information Systems

Abstract: Recently, a new distributed system called a 'virtual organization' is becoming to appear. Such a system is, in general, composed of several autonomous members and these members need to manage and utilize a global database, so-called a Data-Warehousing, as a representation of their global activities. To build an effective Data-Warehouse in this situation, this paper proposes a new design principle that such a global database must be modified and be maintained in an optimal state; also, this maintenance must be made autonomously by the autonomous members. After discussing this principle, this paper describes its implemtation model by using **Bulk-Access Transactions** on behalf of the autonomous members.

1 はじめに

最近、並列データベースシステムの利用方法の一つとして いわゆるデータウェアハウスや仮想企業向けのデータセンタが注目されている [1,2]。一般に データセンタとは、 図1に示した構成の集中型データベースサーバであり、その利用形態は次の三項目にまとめられる：

・[前提1] 自律分散的に活動している参加メンバ 1,..., メンバ N から構成された分散組織において、 組織全体に関する大域的な情報を記述するデータベース （図1のDBG）を 一つのデータベースシステムとしてデータセンタに集中管理する。

・[前提2] 各メンバからは その行動記録データ（ログ）や要求データがデータセンタへ報告される。（この報告は、ログファイルを使ったバッチ処理としてメンバごとにデータセンタへ行なわれることが多い。定期的なバッチ処理として報告する例が代表的である [1]。）データセンタは こうしたデータに基づいてDBG の最新性や一貫性、的確さを維持する。ログデータ自体もDBG の一部 (図1のLOGDB) としてデータセンタに蓄積される。

・[メンバからのデータセンタの利用方法]
各メンバ i は データセンタに対し、DBG と LOGDB とを用いた情報分析・加工操作 (TRANi) を行ない、自分にとって適切な情報をデータセンタから取りだし、その結果に基づいて行動する。ただし、 TRANi は メンバごとに バッチ・トランザクション、または オンライン・トランザクションとしてデータセンタへ要求・実行される。□

一般に データセンタに集めたデータベースの利用方法は データウェアハウス [1,6] と呼ばれる[1]。その代表的な事例には、ログをデータセンタに蓄積しメンバから問い合わせ処理を行なう利用形態、いわゆる OLAP （On-Line Analytical Processing [2]) がある。しかし、上の[前提2] で示したように、自律分散型組織向けのデータセンタにとっては、「大域的な情報を表すデータベースの適切さを維持し、メンバごとに自律的にその大域的情報を活用できる」という機能が中心的な役割を果たしている。そこで本稿では、この「大域的データベースの最適性維持」という機能を より積極的に利用することにより、自律分散型組織にとって有用なデータウェアハウスが構築できることを述べる。 具体的には、「各メンバごとに自律分散的に大域的データベースの最適性を維持する」という枠組を用いて 自律分散組織にとって有用なデータウェアハウスを構築する。

本稿で提案するデータウェアハウス （すなわち、データセンタの利用方法）の具体例としては、例えば 以下に

[1]本稿では、「データウェアハウス」という用語を 「データセンタに置いたデータベースの利用形態や利用方法」という意味で使う。その利用形態を実装したデータベースサーバのことは データセンタと呼んでいる。

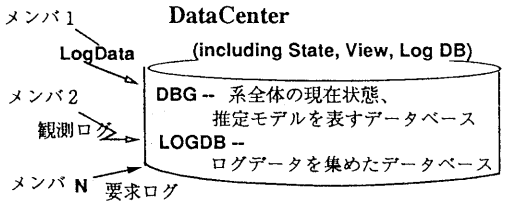


図 1: データセンタの構成

示すような事例があげられる。すなわち、以下に示す事例は いずれも 各メンバが 情報分析だけではなく更新処理も 集中化データベースに対し行なうような場合であり、従来の OLAP とは異なった新しい利用形態である：

(事例1) 図1において、データセンタに 系全体に関する推定モデルを DBG として維持し、メンバからのログ報告に応じて DBG を適切に更新するような利用形態。
→ 例えば、図1を 「分散してマーケティング活動を行なっている店舗群のデータセンタ」とし、そこに 「全地域にまたがったグローバルな 時間・地域・品目・客層別の顧客需要量分布の推定モデル」を DBG として集中管理する、という利用形態が考えられる。（DBG としては 適切な履歴データ集合を用いる。）こうした大域的な推定モデルがデータセンタにあると 組織の戦略決定に有用である。この利用法では、DBG （推定モデル）の適切さをメンバからの報告ごとに維持することが重要である。そのためには、各店舗メンバは 自分の売り上げログデータを定期的にバッチ処理としてデータセンタへ報告し、同時にこのログを元に DBG の一部を より的確・最新なものへと 修正していく必要がある。さらに、各メンバの自律性を保つためには、この DB 処理は メンバごとに別々の処理単位として並行実行されなければならない。（図2にこの事例1を図1において実行する様子を示した。詳細は2節で述べる。）

この事例1は、メンバごとにその活動ログや観測ログがデータセンタへ報告し、そのたびに 大域的推定モデルを更新するモデル、として一般化できる。 このモデルは、ログデータの蓄積を中心にしたデータセンタにとっては有用な利用形態である。

(事例2) 図1において、データセンタに 系全体のグローバルな現在状態を DBG （図1）として管理し、その最適性を系の変化に応じて維持する場合。→ 例として、図1が在庫センタを地域別に分散させた物流システムのデータセンタとし、 そこに全地域にわたる在庫データや出荷可能データの現在値を DBG として集中管理するような場合を考えよう。このとき、在庫管理メンバを複数個用意し、各々が （定期的なバッチ処理として）自分の推定データに基づいて DBG のある領域の最適在庫量を管理すると、DBG （在庫データ）全体の最適性が

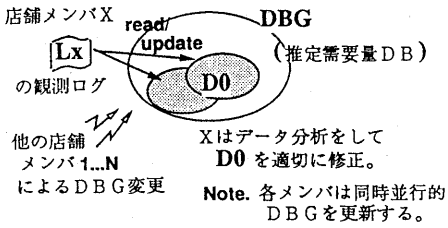


図 2: 大域的推定モデルを維持・利用する場合

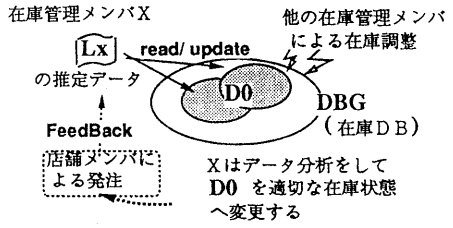


図 3: 大域的な現在状態 DB を維持・利用する場合

自律分散的に維持できる。 (図3に この場合のデータセンタにおける処理の様子を示した。詳細は2節で述べる.) 事例2は、分散組織における大域的状態を表すデータベースの最適性を複数メンバに分散管理させるような場合といえる。

上にあげた二例はいずれも「分散型組織の大域的な現在状態データ、推定モデル、ログデータをDBGとしてデータセンタに集中管理し その最適性をメンバごとに自律分散的に維持する」という枠組として一般化できる。本稿ではこの枠組を仮想組織向けデータウェアハウスの構築モデルとして提案する。以下では このモデルに沿った有用な事例をあげ、その実現に関わる諸問題を論じていく。

以下、2節で 我々が提案するデータウェアハウスのモデルを概説し、その適用事例を三つ述べる。3節では これらの事例を元に2節で提案したデータウェアハウス・モデルをまとめ、その実装方法と並列データベースシステムのシミュレータ上に構築した事例を示す。最後に4節で本稿を要約する。

2 データウェアハウスの構築モデル

2.1 提案するデータウェアハウスモデルの概説

この節では 1節で概説したデータウェアハウスのモデル、すなわち「各メンバが DBG (図1) の最適性や制約を自律分散的に維持する」という枠組に沿って 実際にDBG を利用する形態を論じる。はじめに、我々が提案しているデータセンタの利用形態を次の二項目にまとめておく：

1： データセンタに置くデータベース DBG は 次の三種類に分類される (図1参照)。すなわち、系全体に関する行動推定モデルを表すもの、系全体の現在状態を表すもの、各メンバの活動記録ログデータを集めたもの (図1の LOGDB) の三つである。 以下では この三種類それぞれを利用するデータウェアハウスのモデルを述べる。 DBG には それ自体に必要な制約条件や適切さの判定基準、更新の際の副作用など付帯条件が与えられることが通常である。本稿でもこれらが図1の DBG に設定されていることを前提とする。

2： 我々は、「図1において 各メンバがそれぞれ独自にデータベース操作を実行し、それによって DBG の最適性をメンバごとに自律分散的に維持する」という枠組をデータウェアハウスの構築方針として提案する。1節の事例1、2の実行モデル (図2, 3) からもわかるように、この枠組では、個々のメンバが行なうデータベース処理は 一つのトランザクションとして実行される。当然、各トランザクションは DBG の制約条件を維持した上で より適切になるように DBG を更新しなければならない。また、これらトランザクション群はメンバ間では同時並行的に実行される。

上記二点をまとめると、我々が提案するデータセンタの利用形態とは、 「図1においてメンバごとに トランザクションを実行して データセンタの大域的データベース **DBG の最適性を維持し、メンバ間の自律分散性を損なわないようにしよう、**」 ということに他ならない。このとき、1節の事例1、2を見てわかるように、メンバごとのトランザクションは複雑な情報分析と DBG への大量データ更新とを使って DBG を更新するような処理単位である。我々はこうした処理単位を「大量データをアクセスするトランザクション (Bulk-Access Transaction, **BAT**)」と呼んでおり、従来からその高効率な処理方式を提案している [3,4]。従って、本稿で提案する枠組とは、自律分散メンバごとに **BAT** を用いてデータウェアハウスを構築すること、とも言える。

以下、上述した枠組に沿って上の項目1にあげた DBG 三種類を利用するデータウェアハウスのモデル1、2、3を述べ、その実現方法を論じる。

2.2 モデル1 - DBG が大域的推定モデルを表す場合

最初にあげるデータウェアハウスのモデルは、DBG が大域的推定モデルを表す場合の利用形態である。この利用形態 (以下、モデル1と記す) の例としては、例えば**1節の事例1**のように 地域分散した店舗群向けデータセンタに 顧客需要の予想量などの大域的推定モデルを維持、利用する場合がある。 図2に、事例1において各店舗メンバがデータセンタで実行するデータベース処理 (DB 処理) を示した。図中、DBG は 分散店舗群全体

にわたる特徴別の需要量推定モデルである。1節で述べたように、各店舗メンバは 自分の売り上げログをデータセンタへ定期的に報告し、そのデータに基づいてDBGをより的確に更新しなければならない。 こうした大域的推定モデルがあると、分散型組織全体の戦略決定や 個々のメンバの戦略決定にとって 非常に重要であることは明らかである。

モデル1を実現するには、DBGの作り方、各メンバが行なうべきDB処理、および 複数メンバ間で推定モデルを作り直すやり方、の三点が問題になる。以下では事例1を使って この三点を詳述する。

(1) DBGの作り方： 事例1において、DBGは ある一定期間の推定需要量を 品目、客層、時間帯、地域、などの様々な特徴別に分類して表示したデータベースである、としておく。 具体的には、全店舗が一カ月 (month) 単位で営業活動を行なうような分散組織であるとして、DBGとしては現在の一カ月における特徴別の需要量の推定データをデータセンタに用意したい。 ここではDBGのスキーマを [地域、品目、客層、時間帯、推定需要量] とし、そのタプルは [地域、品目、客層、時間帯] の値に対する総需要量の推定値を示す、と設定しよう。（実際には、このDBGとして 適切なログデータ集合を選んで用いる。） こうすると例えば、'96年3月期一カ月の営業活動においては、DBGは '95年3月期一カ月分の店舗別売上ログデータを そのまま DBGの初期値として使うことにし、店舗メンバから現在の売上ログ報告がくる度に DBGの値をより正確になるよう変更していけば 良い。 すなわち、各メンバは 適当なトリガ条件で (ex. 一日ごと、または ある状況が発生したときに) 自分の売上ログをデータセンタに報告し、DBG (＝現在営業月一カ月間の総需要量推定データベース) を更新する。このDBGを用いると、各メンバは ログ報告のあった範囲内での大域的な情報分析が可能になる。

上述した例では、DBGは 一カ月間のログデータの集まりをそのまま (つまり集約演算などで加工せずに) 使っている。この例から明らかなように、事例1の利用法においてはDBGは ログデータ集合から構成された大規模なデータベースである。

(2) 各メンバのDB処理: 事例1においては 各店舗メンバは 自分の売り上げログをデータセンタへ定期的に報告し、そのデータに基づいてDBGをより的確に更新しなければならない。このDB処理は メンバごとのバッチ処理として実行されるものであり、具体的には 「メンバXがDBGのある領域のデータ分布 $D0$ について 自分のログデータとの間で選択・結合・集計演算を使って情報分析を行ない、その結果に応じてより適切な分布 $D1$ に $D0$ を変更すること」といえる (図2参照)。 このDB操作は、次の三手順から構成されたトランザクションである：

手順1: メンバXは DBGのうち 自分の売上ログ Lxと関連のある領域 $D0$ のデータを選択演算で取り出す。図2では $D0$ としてデータ集合1 $D0[1]$ とデータ集合2 $D0[2]$ とを 取り出している。

手順2： 次に、取り出したデータ集合と Lxとで結合演算、集計処理をして $D0$ のデータ分布の適切さを判定する。（式 $T = Lx \bowtie D0[1] \bowtie D0[2]$ を計算して Lx との照合度が一定値以下であるようなタプル集合を $D0$ から選び出す。照合度とは、例えば、Lxで観測された需要量の分布に対し $D0$ を母集団と見立てた時の類似度の基準を言う。）

手順3： $D0$ のデータ分布を適当な基準に従って更新し、T における照合度を上げる。（例： 上で見つけた $D0$ の不適切なタプル集合に相当する需要量を、Lx のデータ分布に応じたDBGの領域の需要量へ 移動する。） □

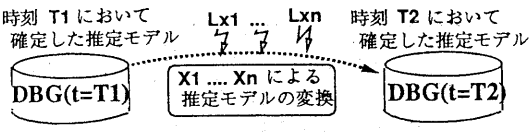
各メンバは 上述したトランザクションを 自分の適当なトリガ条件で起動し、データセンタへ投入する。メンバ間では これらトランザクションは同時並行実行される。また、本質的に 個々のメンバ処理は そのログLxがある程度蓄積されてから起動されるため、各トランザクションは メンバ対応のバッチ処理である。

また、推定モデルDBGを変更する際には、メンバごとの自律的な行動戦略、リスク分析、それへの対策などの意思決定要因が 新しいDBGの推定値を決定する要因になることが多い。そのため、上のトランザクションの手順2、3において、 こうしたメンバごとの意思決定要素やリスク分析・対策をメンバから指定させる、といった要素を加えると より現実的である。（例えば、手順2においてメンバXの報告によって 将来 発生しうる需要量変動の可能性を DBGの現在状態から推定し、その中で ある行動戦略をXが選ぶことで より確実な値に DBGを更新する、などがあげられる。）

このように、メンバごとに大域的推定モデルの適切さを維持するようDBGを更新するという方針に立つと有用なデータウェアハウスが構築できる。この事例1のような場合、以下の二点が特徴といえる：

特徴1： 大規模なログデータを使って一定期間における大域的推定モデル DBG をつくっている。そのため、DBGは場合によっては 100万件規模のログレコードから構成された大規模データベースにもなりうる。このDBGの最適性をメンバごとにトランザクションを使って維持している。（できるだけ集計加工していない形で推定モデルを維持・利用することにより、生データに近い範囲での情報分析ができるはずである。）

特徴2： メンバごとのトランザクションは DBGへ Complex Queryを使って複雑な情報分析を行ない、大量データ更新操作を使って DBGのある領域をより適切になるよう更新する。こうしたトランザクションはファイルスキャンを主に使って大量データの read, update を行なう。我々は こうした 大量データをアクセスするト



X1, X2, ... , Xn は制約条件の下で **DBG(t=T1)** を **DBG(t=T2)** へ変更する。 **Xi** のログ報告 **Lxi** により **DBG** の制約違反部分はバックトラックして修正される。

図 4: 観測期間 [T1, T2] における推定モデルの変更方法

ランザクションを BAT (Bulk-Access Transaction) と呼び、従来の少数レコード処理のトランザクションと区別する。

(3) DBG の再更新のやり方: 上の二点の他に、モデル 1 では あるメンバ X が書き換えた DBG の値を 別のメンバ Y がさらに更新する際、どう更新するか、という問題がある。 具体的には、メンバ X のログによって更新された新しい推定モデル DBG の値を別のメンバ Y によって さらに変更する、または X による変更を修正する、という状況が発生する。

一般には この再更新のやり方は DBG 自体の制約条件として事例ごとに与え、制約に違反した DBG の値はバックトラックして再更新できる、と設定する必要がある。 (すなわち、各メンバが発行する BAT は、そのメンバ固有の DBG 更新の他にも DBG の与えられた制約に従った副作用的な DBG の更新や、上で述べた「他メンバの更新部分のバックトラック」をも行なわねばならない。)

モデル 1 の場合、この「DBG 再更新問題」は次のように形式化できる:

[DBG 更新モデル]: ある観測期間 [T1, T2] において その期間内に発生するログ報告 Lx1, Lx2, ..., Lxn 各々に対し推定モデル DBG を更新し続ける場合を考えよう。 (時刻 t= Ti における DBG の値を DBG(t=Ti) と表記する.) このとき、DBG(t=T1), DBG(t=T2) は 確定した推定モデルであり、一方、期間 [T1,T2] の間に ログ報告を受けて修正されている DBG の値は未確定なものである、と見なすことにする。 こうすると、 メンバ X1,..., Xn による推定モデル DBG の再更新問題とは、DBG(t=T1) の値から DBG(t=T2) への写像を X1,...,Xn で 交渉して決める分散制約充足問題、といえる。

上述した DBG 更新モデルを 図 4に 示した。図中、DBG(t=T1), DBG(t=T2) が 確定した推定モデルである。図に示すように、観測期間中に 各メンバ Xi からログ報告 Lxi が発生すると、それに応じて 推定モデル DBG が修正されていく。このようにして、新しい推定モデル DBG(t=T2) が DBG(t=T1) から全メンバによって作られていく。ただし、あるメンバ Y がログ報告を行なったとき、与えられた制約に違反した DBG の領域の値については、別メンバ X が 既に DBG(t=T2) 向けに変更したデータであっても Y がそれを修正できる、と設定する。 正確には、制約に違反した DBG の領域の値は その初期値 (DBG(t=0) での領域の値) へバックトラックし、再び DBG(t=T1) から DBG(t=T2) の値への推定のやり直しとなる。 (時刻 T2 になると DBG(t=T2) は確定し、それを元に次の推定モデル DBG(t=T3) の作成が開始されることになる。)

これ以外の DBG 再更新の方法としては、メンバのログ報告が出るごとに DBG の値を 確定した推定モデルとして扱うこともできる。この場合、DBG の再更新問題は 一般的なルールベースによる推定モデル管理問題であり、事例ごとに再更新ルールを記述することになる。

以上述べてきたように、図 2の形式で メンバ間の DB 処理を同時並行実行し、図 4の方針で推定モデルを修正する利用形態がモデル 1 である。このモデル 1 は、ログデータ追加に伴って推定モデルを修正・維持する問題に他ならず、データマイニング的な使い方に適している: 例えば、推定モデル DBG としてログデータベースを用い、ログ報告の度に DBG に対する分類木の詳細化を行なったり ルールの検出・修正を行なう、などの事例も容易に実現できる。したがって モデル 1 は今後きわめて重要なデータウェアハウスモデルといえる。

2.3 モデル 2: DBG が 大域的な現在状態を表す場合

次に、「分散型組織の大域的な現在状態を DBG とし、その適切さを複数のメンバによって自律分散的に維持する」利用形態を述べる。この利用形態は、**1 節の事例 2** のように分散在庫システムの在庫状況を DBG として表し、その最適性を複数の在庫管理メンバによって分散管理する例にあたる。

図 3に 事例 2 の実現モデルを示した。図中、DBG は地域分散した供給所の現在の在庫状態を表しており、データセンタにある。この図では DBG の適切さを在庫管理メンバ 1,.., N によって分散管理している。すなわち、各在庫管理メンバは DBG の ある領域 D0 における在庫最適性を (定期的などのバッチ処理として) 判定し、より的確な在庫量分布 D1 へと変更する (ようするに、モデル 1 と同じく D0 の適切さを 自分のログ Lx と合わせて データ分析、判定を行ない、その結果によって D0 の変更、すなわち適当な在庫調整や新規生産を指示する。)。

この事例 2 では、DBG は モデル 1 と同じく特徴別の現在在庫量を表すデータベースとする。最近では 一カ月単位での細かい特徴別の在庫データ管理は珍しくなく [1]、こうした現在状態を表す大規模なデータベースに対し最適性管理を行なう機構は重要である。こうした大規模な在庫データベースに対し各メンバが行なう処理は、上述したようにデータ分析と大量データ更新を伴ってお

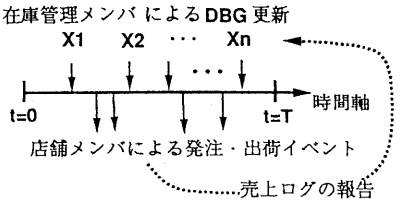


図 5: 期間 [0,T] における DBG の最適性管理の運用

り、やはり BAT としてモデル化できる。

一般に在庫管理メンバが担当する DBG の領域はメンバ間で重複しているが、各メンバ X は各々独自のデータ分析をして （メンバ間で衝突しているような）担当領域を変更する。 こうした再書き換えのやり方は モデル 1 と同様に行なわれる： すなわち、一般には DBG の制約条件として 事例ごとに設計者が与える必要がある。例えば、この事例 2 では 各メンバ X が調整した在庫量は 他の在庫管理メンバからは一定期間変更しない、を原則にする。ただし、DBG には 制約条件を与え、メンバ X から見て制約を満たさないような在庫量は 一定範囲で再更新できる、とする。

これと並行して 図 3 では定期的に DBG から店舗メンバによるバッチ形態の出荷 (DBG への delete) が行なわれ、それに伴って店舗メンバの活動ログデータが各在庫管理メンバの推定データへ （フィードバックとして） 追加されていく。この出荷処理自体も 別の トランザクションであり、一般には BAT として実行される。こうしたフィードバック的な状況変化もこの利用形態では簡単に扱える。

例： DBG の最適性管理の運用例について： 事例 2 における DBG の最適性管理の様子を図 5 に示した。図中、一定期間 [0,T] において在庫管理メンバ X1,...,Xn がそれぞれ自分が決めたトリガ条件に従って起動され、その時点での DBG の最適性を判定し 適切に在庫状態を変更している。ただし、一度 あるメンバ X が更新した特徴別の在庫量自体は区間 [0,T] においては変更されない、と設定する。 (i.e. 他メンバ Y によって同じ特徴値における在庫量が追加されることはあるが、Y からは X が既に確保した在庫量は変更されない。ただし、Y の在庫量判定のときに DBG の制約を満たさない在庫量については Y が再調整できる、と設定する.) 区間 [0,T] での在庫調整がおわると、次の区間 [T,2T] において DBG の値は 全メンバによって再度調整される。 □

以上述べてきたように、モデル 2 は 図 3 に示した四つの要素 (i.e. 制約管理メンバ間での分散最適性管理、複数の制約管理メンバによる書き換え条件制約の下での DBG の再更新、外部事象を表すメンバによる DBG への変更、および それに伴う制約管理メンバへのフィードバック、の四つ) から構成される。モデル 2 は、外部事象により変更され続ける一般的なデータベースについて、その最適性を変更分からフィードバックしながら分散管理するモデルといえる。

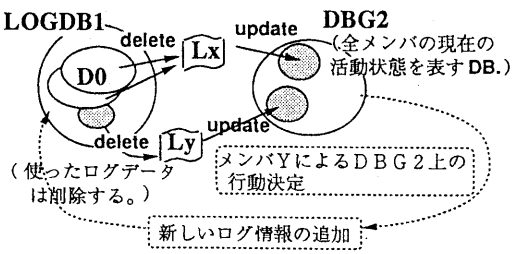


図 6: ログデータベースを主に利用する場合

2.4 モデル 3：ログデータベースを使う場合

三つ目のデータウェアハウス・モデルとして、ログデータベースを利用する場合を考えよう。図 6 にその一般的なモデルを示した。

事例 1 と同じく地域分散した店舗群向けのデータセンタを例に用いてこの図を説明する。 今、全店舗メンバから報告される売上ログを 一つにマージし特徴別に分類したデータベース LOGDB1 （図 6) をデータセンタで管理する、としよう。さらに 全店舗の営業活動状態を 別の大域的現在状態データベース DBG2 として データセンタに維持しておく。このとき、モデル 3 として、「各店舗メンバ X は LOGDB1 と自分の現在状態データ (Lx) との間で集計分析を行ない、その結果 自分に より有利なマーケット領域に対して営業活動を行なう」という利用形態が考えられる。すなわち図 6 において、X は 自律的に起動して LOGDB1 と Lx とでデータ分析した後に自分の行動戦略を決定し、それに応じて DBG2 における自分の出荷分布を変更する。

このメンバ X の DB 処理は メンバごとに定期バッチ処理として実行されるトランザクションであり、Complex Query によるデータ分析と LOGDB1、DBG2 への大量データ更新を伴うような BAT である。ただし X は、DBG2 の変更に際しては 他メンバと DBG2 上の制約を越えた競合は避け、LOGDB1 でも、自分が使用したログデータはマークをつけ、他メンバと重複して利用しない、とする。

このように、図 6 に示した利用形態 （モデル 3） は、集めたログデータを複数メンバが自律的に仕分けして各自のマーケッティング行動を改良していく時に有効なモデルである。 (DBG2 による活動の結果、新しいログが LOGDB1 に追加されていくので フィードバック的な現象にも対応している。) モデル 3 は 従来の全メンバ一括のバッチ処理業務 （ログデータのバッチ処理） をメンバごとに分割し、メンバごとに自由に BAT を使って実行することに相当する。

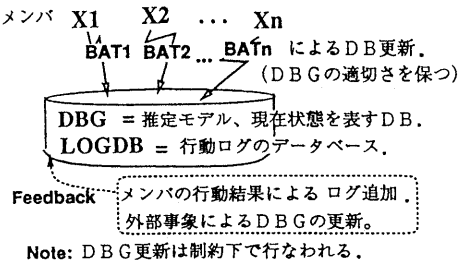


図 7: データウェアハウスモデルの一般形

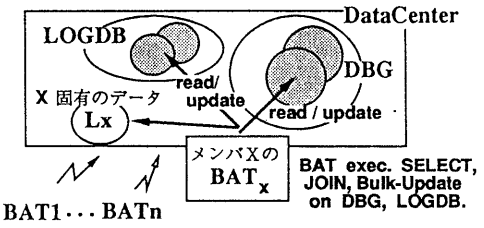


図 8: BAT によるデータ倉庫の処理モデル

このモデル3とモデル1や2との違いは、ログデータベースの上で メンバ間で書き替え（正確には重複利用を避けるための delete 処理）が発生する点である。モデル3では、LOGDB へは 時間経過に従ってログ追加が生じるだけであり、モデル1や2のような面倒な DBG 書き替え制約条件は LOGDB 更新に対してはいらない。

3 BAT を用いたデータ倉庫の実現

3.1 提案モデルのまとめ

2節では 「自律分散メンバによる DBG の最適性維持」という枠組を提案し、それに沿って有用なデータウェアハウスの構築モデル（モデル1、2、3）をあげた。これらの事例から提案した枠組の有効性が判る。

2節で述べた一般的なデータウェアハウスの構築モデルを図7にまとめて示す。また、この構築モデルをデータセンタ上でデータベース処理する際の実行モデルを図8に示す。この二つの図が我々が提案するデータウェアハウスモデルであり、次の4点がその特徴である：

＃1： データセンタには ログデータベース LOGDB と 推定モデルや現在状態を表すデータベース DBG とが置かれ、各参加メンバが自律分散的にその適切さを維持・利用する。DBG はログデータで作られる場合（モデル1）もあり、LOGDB と共に一般には大規模なデータベースである。（図7, 8参照。）

＃2： モデル1、2、3全てにおいて、各メンバが行なう DB 処理は「DBG、LOGDB への（Ad-hoc で複雑な）データ分析と大量データ更新を持つトランザクション」である.（図8参照)。こうしたトランザクションはファイルスキャンを主に使って大量データの read/update を行なっており、Bulk-Access Transaction (BAT) としてモデル化できる [3,4]。 したがって、各メンバは 自分がやるべき DB 処理を BAT として発行し、これら BAT をデータセンタ上で同時並行実行することで DBG の適切さを自律分散的に維持している。（図8では 各メンバの BAT は DBG の最適性維持の他に DBG 自体の制約維持や副作用的な DB 更新も行なうことになる。）また、各メンバは 相異なるデータ分析を行なうが、更新する DBG の領域はメンバ間で衝突する。そのため相異なる BAT の間にはファイル単位ロックなどによる同時並行制御が必要である。

＃3： 一般には、他メンバが更新したデータを再更新できる範囲は DBG 自体の制約条件として事例ごとに決める必要がある。（モデル1ではこれをある確定した DBG から別の確定した DBG への写像を決めるような制約充足問題、として推定モデルを管理した。モデル2でも同じく書き換え制約や一定時間内の更新禁止などを用いた。モデル3ではログデータに対し重複利用禁止という制約を使った。）

＃4： 各メンバのトランザクションは メンバごとに任意のトリガ条件でバッチ処理としてデータセンタへ投入され、他メンバの処理と同時並行実行されることになる。モデル2、3では さらに DBG に基づいたメンバの行動の結果、フィードバック的に データセンタへログデータや状況データの追加更新が行なわれている。（図7, 8参照).

上述したように、提案方式ではメンバごとに BAT を実行することで 各メンバの自律性を保ちながら DBG の適切さを維持しており、システムの運用中に予測不可能に発生する状況変化にも迅速に対応できる。この点で 図8, 7の提案モデルは 従来の全メンバー括のバッチ処理とは異なっており、分散型組織に適したデータウェアハウスといえる。

3.2 実装例

本稿で提案したデータウェアハウスの自律分散的な最適性維持を実現するためには、BAT の同時並行処理をデータベースサーバで行なう必要がある。並列データベース (DB) システムにおける BAT の同時並行処理方式は 著者が既に提案しており [3,4]、この節では 文献3の BAT 並行制御方式に基づいて図8のデータウェアハウスを実装した例を示す。

疎結合型並列 DB サーバ上で BAT 一つを実行する際の様子を図9に示した。図中、DBG を構成するファイル (図中の A,B,...,Y) は並列 DB サーバの計算機ノードへ分散配置されており、BAT 一つは 適当なファイルを

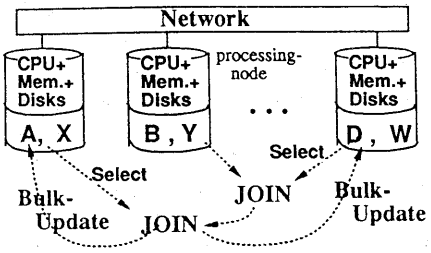


図 9: BAT の並列 DB サーバ上の実行モデル

選択・結合して 大量データ更新を行なっている。この形の BAT を複数個 同時並行実行するために、文献 3 では直列可能判定グラフを使ったスケジューラにスケジュールコストの評価機構を導入して BAT 間のロック競合・リソース競合を削減する方法を提案している。

このスケジューラを用いて、2.2 節 モデル 1 で述べた事例 1 の推定モデル管理を並列 DB サーバのシミュレータ上に実現した。付録 1 にその実行の様子を抜粋して示した。この付録からも、モデル 1 が 複数メンバによるデータマイニング的な事例に相当することが判る。[2]

4 おわりに

本稿では 自律分散型組織向けのデータウェアハウスをメンバごとに **BAT** (大量データをアクセスするトランザクション) を実行することで実現する方式を述べた。提案したデータウェアハウスの構築モデルとは、すなわち、データセンタに置いた集中データベースの適切さをメンバごとに自律分散的に維持すること、である。本稿では この構築モデルに沿った事例として、データベースとしてメンバ全体に関する大域的推定モデルを自律分散的に維持する場合、 系全体の大域的な現在状態を維持しその適切さを分散管理する場合、 ログデータベースをメンバごとに使って大域的に見て的確な行動を取る場合、にわけ その詳細な実現方法を述べた。

現在のデータウェアハウスでは OLAP など問い合わせ処理が中心だが、本稿で述べた自律分散的なデータウェアハウスの最適性維持を考えると BAT の並行処理が今後のデータベースサーバには重要な機能といえる。並列データベースシステムにおける BAT の同時並行処理方式は 著者が既に提案しており [3]、本稿では この方式を用いて 大域的推定モデル維持の場合の実現例を示した。

本稿で提案したデータウェアハウスは 推定モデルや現在状態を表すデータベースに対する複雑なビュー維持機構ということもできる。こうした処理は従来は 全メンバ一括バッチ処理として扱うことが多い。しかし、自律分散型組織向けのデータウェアハウス応用を考えると、本稿で提案した利用形態が今後増えてくると予想される。特に、モデル 1 (大域的推定モデルの維持) の場合は データマイニング機構としてのデータセンタ利用であり、今後の 高並列データベース計算機にとっては主要な応用分野となると期待している。

[2]付録 1 のデモでは、DBG は 大量のログレコード集合から構成された推定モデルであり、8 地域、4 品目グループに渡る推定需要量データベースである。(DBG は 2.2 節 事例 1 のスキーマに従って作っている。) また、付録 1 では 推定モデル DBG を表示する際に 8 地域・4 品目グループに推定需要量を集計した View を用いているが、DBG 自体は ログレコード数万件を集計加工せずにそのまま使って構成している。 付録 1 においては、店舗メンバ X (X=11) からログ Lx が報告されるとそれに応じた BAT が起動されており、DBG において Lx と影響のある領域 (Area0,5) のデータ分布が変更されていく。(あわせて、変更の際には メンバ X によるリスク分析と今後の行動戦略の決定を X は行なっている。) その後、X の戦略決定の結果 DBG のうち修正すべき推定モデルの部分が判定され、元の値にバックトラックされている。

参考文献

1: Inmon, 初めてのデータウェアハウス構築. トムソン出版.1995.
2: Codd 他、*Providing OLAP to User-Analysis*, Codd&Associates.1995.
3: 大森. 他. Scheduling Batch Transactions on Shared-Nothing Parallel Database Machine. IEEE 7th Data Engineering '91. 1991.
4: 大森. Concurrency Control of Bulk-Access Transactions on Shared-Nothing Parallel Database Machine. 東京大学大学院情報工学専攻博士論文. 1990.
5: 大森. 情処全国大会第 52 回 2P-3.
6: Silbershats, etc. ed. Database Research: Achievements and Opportunities Into the 21th century. *Report of NSF workshop on the Future of Database Systems Research, 1995 May.*

付録 1 : モデル 1 (大域的推定モデル維持) の実行例.

```
現在の推定需要量モデル DBG
  (8 地域 A0 - A7. 4 品目グループ G0-G3 の集計値表示)

  [現在の DBG の値]
AreaID  A0  A1  A2  A3  A4  A5  A6  A7
G0 = [  82  64  52  42  52  64  84  64 ]
G1 = [  55  47 114 104 114  47  57  47 ]
G2 = [  42  32  75  55  75  65  55  58 ]
G3 = [  60  60  37  37  37  37  37  37 ]
                    ↓
Event: メンバ 11 が Area0,Area5 に関するログを報告し、
  DBG のうちこの Area のデータ分布を更新しようとする.
                    ↓
 [メンバ 11 によるリスク評価と対策の選択]
Warning: Area5 において推定需要量 10 減少の可能性あり.
    ==> メンバ 11 は 特に対策はしない、とする。
    ==> OK: G1 has lost 10-resources on Area5.
                    ↓
 [メンバ 11 の戦略による副作用的な DBG の更新を行なう]
Positive FeedBack on Area5....
                    ↓
 [メンバ 11 が更新した後の DBG は:]
G0 = [  82  64  52  42  52  57  84  64 ]
G1 = [  55  47 114 104 114  37  57  47 ]
G2 = [  40  32  75  55  75  48  55  58 ]
G3 = [  83  60  37  37  37  29  37  37 ]
                    ↓
 [BackTrack: メンバ 11 のログ報告の結果、Area0,5 で
  推定モデルの制約違反を検出し、 DBG を再修正する.]
      16 resources are backtracked from G2 to G1.
                    ↓
 [バックトラック後の DBG の値は:]
G0 = [  82  64  52  42  52  57  84  64 ]
G1 = [  71  47 114 104 114  53  57  47 ]
G2 = [  24  32  75  55  75  32  55  58 ]
G3 = [  83  60  37  37  37  29  37  37 ]
                                  (付録おわり)
```